TUGAS AKHIR – KI141502

MIXED REALITY MENGGUNAKAN GOOGLE CARDBOARD UNTUK PENATAAN RUANGAN DENGAN OBYEK TIGA DIMENSI

MUHAMMAD GHOZIE MANGGALA
NRP 5112100093

Dosen Pembimbing
Darlis Herumurti, S. Kom, M. Kom,
Anny Yuniarti, S.Kom., M.Comp.Sc.

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2016

(Halaman ini sengaja dikosongkan)

TUGAS AKHIR - KI141502

MIXED REALITY MENGGUNAKAN GOOGLE CARDBOARD UNTUK PENATAAN RUANGAN DENGAN OBYEK TIGA DIMENSI

MUHAMMAD GHOZIE MANGGALA
NRP 5112100093

Dosen Pembimbing
Darlis Herumurti, S. Kom, M. Kom,
Anny Yuniarti, S.Kom., M.Comp.Sc.

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2016

(Halaman ini sengaja dikosongkan)

FINAL PROJECT- KI141502

MIXED REALITY USING GOOGLE CARDBOARD FOR ROOM ARRANGEMENT WITH THIRD DIMENSION OBJECT

MUHAMMAD GHOZIE MANGGALA
NRP 5112100093

Advisor
Darlis Herumurti, S. Kom, M. Kom,
Anny Yuniarti, S.Kom., M.Comp.Sc.

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY
SEPULUH NOPEMBER INSTITUTE OF TECHNOLOGY
SURABAYA
2016

(Halaman ini sengaja dikosongkan)

LEMBAR PENGESAHAN

MIXED REALITY MENGGUNAKAN GOOGLE CARDBOARD UNTUK PENATAAN RUANGAN DENGAN OBYEK TIGA DIMENSI

Tugas Akhir

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Rumpun Mata Kuliah Interaksi, Grafika, dan Seni
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh:
MUHAMMAD GHOZIE MANGGALA
NRP. 5112 100 093

Disetujui oleh Dosen Pembimbing Tugas Akhir:

Darlis Herumurti, S. Kom, M. Kom. .....
NIP: 19771217 200312 1 001 (pembimbing 1)

Anny Yuniarti, S.Kom., M.Comp.Sc. .....
NIP: 19810622 200501 2 002 (pembimbing 2)

SURABAYA
MEI, 2016

(Halaman ini sengaja dikosongkan)

Mixed Reality menggunakan Google Cardboard untuk Penataan Ruangan dengan Obyek Tiga Dimensi

Nama Mahasiswa : Muhammad Ghozie Manggala
NRP : 5112 100 093
Jurusan : Teknik Informatika FTIf-ITS
Dosen Pembimbing I : Darlis Herumurti, S. Kom, M. Kom,
Dosen Pembimbing II : Anny Yuniarti, S.Kom., M.Comp.Sc.

ABSTRAK

Mebel atau furniture merupakan perlengkapan rumah yang mencakup semua barang yang biasa ditemukan di dalam rumah seperti kursi, meja, lemari dan lain sebagainya.Penataan ruangan dengan cara konvensional membutuhkan tenaga yang lebih banyak karena harus melepas-pasang benda yang ada. Penataan ruangan dengan teknologi komputer dirasa masih memiliki keterbatasan seperti lingkungan yang kurang nyata, interaksi antara obyek dan manusia, dan lain sebagainya.

Hingga saat ini, penelitian mengenai interaksi manusia dan komputer sangat banyak. Beberapa di antaranya menggunakan Head Mounted Display(HMD) yang menampilkan gambar melalui kamera dengan jarak sangat dekat dan sekarang dikenal dengan Virtual Reality, Augmented Reality yang menampilkan benda melalui perangkat, dan lain sebagainya. Mixed Reality sendiri yang merupakan gabungan antara Virtual dan Augmented Reality mencoba menggabungkan antara dunia vitual dan dunia nyata sehingga menghasilkan lingkungan baru.

Penggunaan teknologi Virtual Reality, Augmented Reality, dan Mixed Reality menjadi terobosan baru dalam hal penataan ruangan. Oleh sebab itu, Tugas Akhir ini mengimplementasikan teknologi Mixed Reality. Pengguna dapat berinteraksi dengan obyek virtual menggunakan pointer yang dikontrol dengan tangan. Untuk itu, tugas akhir ini dibangun menggunakan bantuan Unity3D dan Google VR (Cardboard)

SDK serta diimplementasikan dalam perangkat Android. Dengan dibangunnya aplikasi ini, penggunaan energi dalam menata ruangan sangat efektif.

Kata kunci**: **Penataan Ruangan, Virtual Reality, Augmented Reality, Mixed Reality, Unity, Google Cardboard, Cardboard SDK, Android

MIXED REALITY USING GOOGLE CARDBOARD FOR ROOM ARRANGEMENT WITH THIRD DIMENSION OBJECT

Student Name	: Muhammad Ghozie Manggala
NRP	: 5112 100 093
Major	: Teknik Informatika FTIf-ITS
Advisor I	: Darlis Herumurti, S. Kom, M. Kom,
Advisor II	: Anny Yuniarti, S.Kom., M.Comp.Sc.

ABSTRACT

Meubel or furniture is house objects that we usually seen in our home like chairs, tables, cupboards, and others. Conventional room arrangement needs more energy because their users need to plug and unplug the house objects. Room arrangement with computer technology still having more limitation, like their reality-less environment, human-computer interaction, and many others.

Until this day, there are many researchs about human and computer interaction. Some of them were Head Mounted Display(HMD) that show an image in front of users bare eye (now its called Virtual Reality), Augmented Reality that shows imaginary object in users device, and many others. Mixed reality itself was the mix between Virtual and Augmented Reality that trying to merge virtual world and real world, so it can make a new environment

The use of Virtual Reality, Augmented Reality, and Mixed Reality can be used to do room arrangement. That's why this Final Project is implementating Mixed Reality technology. Usesr can interact with virtual object using pointer that controller by user's hand. So, this Final Project is using Unity3D and Google VR (Cardboard) SDK, then

implemented in Android device. With this application, the use of user's energy for arrangement is very effective.

Keywords**: **Room Arrangement, Virtual Reality, Augmented Reality, Mixed Reality, Unity, Google Cardboard, Cardboard SDK, Android

KATA PENGANTAR

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

Segala puji dan syukur kehadirat Allah Subhanahu wa Ta'ala yang telah memberikan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul "Mixed Reality menggunakan Google Cardboard untuk Penataan Ruangan dengan Obyek Tiga Dimensi".

Pengerjaan Tugas Akhir ini adalah momen bagi penulis untuk mengeluarkan seluruh kemampuan, hasrat, dan keinginan penulis sejak pertama kali memasuki dunia kuliah di kampus Teknik Informatika Institut Teknologi Sepuluh Nopember Surabaya.

Dalam pelaksanaan dan pembuatan Tugas Akhir ini, penulis mendapatkan banyak bantuan dari berbagai pihak. Melalui lembar ini, penulis ingin menyampaikan ucapan terima kasih secara khusus kepada:

1. Allah SWT atas limpahan rahmat dan rezeki-Nya sehingga penulis dapat menyelesaikan Tugas Akhir.
2. Ayah penulis, Nardiyono dan Ibu penulis, Windu Utmi, yang selalu memberikan dukungan, doa, perhatian, kasih sayang, serta modal finansial dalam menjalani kuliah.
3. Bapak Darlis Herumurti selaku dosen pembimbing Tugas Akhir pertama yang juga merupakan Ketua Jurusan Teknik Informatika ITS yang telah memberikan pengarahan dan pembimbingan dalam Tugas Akhir, Kuliah, dan kegiatan lainnya.
4. Ibu Anny Yuniarti selaku dosen pembimbing Tugas Akhir kedua yang dengan sabar membimbing penulis dalam pengerjaan Tugas Akhir ini.
5. Bapak Ridho Rahman Hariadi, selaku dosen wali yang telah berkenan memberikan saran dan arahan kepada penulis selama menjalani studi.
6. Bapak Radityo Anggoro selaku dosen koordinator Tugas Akhir yang telah membantu penulis atas segala sesuatu yang

berkenaan dengan syarat-syarat dan terlaksananya sidang Tugas Akhir.

7. Dosen-dosen Teknik Informatika yang dengan sabar mendidik dan memberikan pengalaman baru kepada penulis selama berkuliah di Teknik Informatika.
8. Staf TU Teknik Informatika ITS yang senantiasa memudahkan segala urusan penulis di jurusan.
9. Rekan-rekan penghuni Laboratorium Komputasi Cerdas dan Visi yang memberikan ruang bagi penulis untuk tinggal selama beberapa bulan.
10. Rekan-rekan dan pengelola Laboratorium Interaksi, Grafika, dan Seni yang telah fasilitas sehingga penulis dapat berdiskusi dengan peserta Tugas Akhir yang lainnya
11. Rekan-rekan dan sahabat-sahabat penulis angkatan 2012 yang telah memberikan dorongan motivasi dan bantuan kepada penulis.
12. Rekan-rekan penulis pada Rumpun Mata Kuliah Interaksi, Grafika, dan Seni yang selalu bahu membahu dalam hal kesuksesan bersama.
13. Pihak-pihak lain yang tidak sengaja terlewat dan tidak dapat penulis sebutkan satu per satu.

Penulis telah berusaha sebaik mungkin dalam menyusun Tugas Akhir ini, namun penulis memohon maaf apabila terdapat kekurangan, kesalahan maupun kelalaian dalam pengerjaannya. Kritik dan saran yang membangun dapat disampaikan kepada penulis sebagai bahan perbaikan selanjutnya.

Surabaya, Juni 2016
Penulis

Muhammad Ghozie Manggala

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

(Halaman ini sengaja dikosongkan)

KODE SUMBER

(Halaman ini sengaja dikosongkan)

BAB I
PENDAHULUAN

Bab ini memaparkan garis besar Tugas Akhir yang meliputi latar belakang, tujuan dan manfaat pembuatan, rumusan dan batasan permasalahan, metodologi pembuatan Tugas Akhir, dan sistematika penulisan.

1.1 Latar Belakang

Mebel atau furniture merupakan perlengkapan rumah yang mencakup semua barang yang biasa ditemukan di dalam rumah seperti kursi, meja, lemari dan lain sebagainya. Mebel merupakan hal yang penting dalam kehidupan sehari-hari. Rumah, sekolah, hotel, dan restoran adalah beberapa bangunan yang menggunakan furniture di dalamnya. Penataan mebel yang baik dan sesuai fungsionalitas menjadi kebutuhan dalam dunia modern ini. Beberapa aplikasi seperti *Roomstyler 3D, Planner5D,* dan *Room Arranger* dapat digunakan untuk mendesain dan menata ruangan beserta mebel secara virtual menggunakan inputan dari mouse dan keyboard. Sayangnya desain tersebut hanya dapat dilihat di dunia maya.

Dari sisi masyarakat umum, penataan ruangan bisa dikatakan masih manual, yaitu langsung melepas dan memasang ke tempat yang disukai. Tentu saja usaha dan tenaga yang digunakan dapat terbuang dengan percuma. Selain itu jika masyarakat ingin membeli produk mebel atau furniture yang baru, mereka hanya bisa datang ke toko mebel dan langsung membelinya tanpa bisa mengetahui seberapa baik dan cocok mebel pada ruangan tersebut. Solusi yang mencoba ditawarkan adalah dengan adanya buku katalog furniture rumah. Namun, masyarakat belum bisa melihat langsung bagaimana penampilan furniture tersebut dengan ruangan di rumahnya.

Desain mebel beserta ruangannya menggunakan teknologi *mixed reality* (*augmented reality* dan *virtual reality*) dapat mempermudah serta mengurangi tenaga yang diperlukan untuk

mendesain mebel dan ruangan. *Mixed reality* merupakan *virtual reality* yang menggunakan lingkungan dunia nyata dan menggunakan *augmented reality* yang ditampilkan dalam *virtual reality* [1].

Penelitian sebelumnya yang dilakukan Taehee Lee dan Tobias Hollerer yang disebut dengan HandyAr memungkinkan pengguna untuk menggerakan benda menggunakan gestur tangan dengan satu kamera saja. Namun, penggunaan tangan pada penelitian tersebut dipakai sebagai marker, bukan untuk interaksi dengan obyek [2].

Dalam penelitian lainnya, Mathias Kölsch menggunakan gestur tangan untuk untuk berinteraksi dengan tombol-tombol virtual. Namun tombol tersebut bersifat statis, mengikuti kamera pengguna [3].

Gerd Bruder dkk meneliti tentang studio virtual untuk explorasi arsitektural menggunakan ruangan hijau dan *Head Mounted Display* (HMD). Penelitian tersebut menggunakan warna yang berbeda antara latar belakang dengan obyek tiga dimensi. Warna tersebut yang nanti akan diganti menjadi tekstur virtual. Namun, penggunaan ruangan virtual dan HMD membutuhkan biaya lebih banyak untuk digunakan secara luas [4].

Dengan demikian solusi yang mudah dan murah adalah menggunakan teknologi *mixed reality* menggunakan perangkat Android dan Google Cardboard untuk menata ruangan.

1.2 Rumusan Masalah

Rumusan masalah yang diangkat dalam tugas akhir ini antara lain:

1. Bagaimana mendeteksi tangan pengguna sebagai kontrol aplikasi?
2. Bagaimana manipulasi obyek virtual dan kontrol aplikasi menggunakan tangan pengguna?
3. Bagaimana menampilkan obyek virtual ke dalam dunia nyata menggunakan Google Cardboard?

1.3 Batasan Masalah

Permasalahan yang dibahas dalam tugas akhir ini memiliki beberapa batasan, di antaranya sebagai berikut:

1. Fokus Tugas Akhir kali ini adalah untuk membuat aplikasi penataan ruangan menggunakan tangan sebagai kontrol aplikasi.
2. Input berasal dari gambar kamera yang didapatkan saat aplikasi berjalan dan obyek virtual tiga dimensi yang telah diunduh bersama aplikasi.
3. Output berupa hasil penggabungan gambar kamera dengan obyek virtual 3 dimensi yang ditampilkan dalam layar *smartphone* berbasis Android.
4. Bahasa yang digunakan adalah bahasa pemrograman C#.
5. Menggunakan *middleware* Unity5 yang mendukung Google Cardboard SDK
6. Pengguna hanya dapat menengok ke kanan, kiri, atas, dan bawah untuk melihat lingkungan ruangan.
7. Warna latar belakang ruangan dan gambar yang ditangkap kamera putih atau memiliki tingkat keabuan di atas 50%.

1.4 Tujuan

Tujuan dari Tugas Akhir ini adalah untuk membuat aplikasi *mixed reality* berbasis Android dengan Google Cardboard untuk penataan ruangan menggunakan gestur tangan yang dapat digunakan oleh masyarakat umum.

1.5 Manfaat

Manfaat dari hasil pembuatan Tugas Akhir ini antara lain:

1. Interaksi dengan teknolgi baru dalam *Virtual Reality* dan *Augmented Reality*.
2. Membuat aplikasi *mixed reality* untuk penataan ruangan.

3. Menghemat tenaga dan usaha yang digunakan untuk menata ruangan.

1.6 Metodologi

Pembuatan Tugas Akhir dilakukan menggunakan metodologi sebagai berikut:

A. Studi literatur
   Pada studi literatur ini, akan dipelajari sejumlah referensi yang diperlukan dalam pembuatan aplikasi yaitu sebagai berikut:
   1. Mixed Reality
   2. Unity3D;
   3. Google Cardboard SDK;
   4. Deteksi ujung tangan

B. Perancangan perangkat lunak
   Pada tahap ini, dilakukan analisa awal dan pendefinisian kebutuhan sistem untuk mengetahui permasalahan yang sedang dihadapi. Selanjutnya, dirumuskan rancangan sistem yang dapat memberi solusi terhadap permasalahan tersebut. Langkah yang akan digunakan pada tahap ini adalah sebagai berikut:
   1. Pencarian dan pendataan materi yang akan digunakan
   2. Perancangan sistem dan mekanisme
   3. Analisis kebutuhan non fungsional.

C. Implementasi dan pembuatan sistem
   Aplikasi ini akan dibangun dengan bahasa pemrograman C# dan Unity3D serta diimplementasikan dalam perangkat Android dengan Google Cardboard. Sedangkan obyek tiga dimensi diunduh dari situs penyedia obyek tiga dimensi gratis.

D. Uji coba dan evaluasi
   Pada tahap ini, akan dilakukan pengujian terhadap perangkat lunak menggunakan data atau skenario yang telah dipersiapkan sebelumnya yakni sebagai berikut:

1. Pengujian *black-box*
   Pengujian *black-box* adalah pengujian yang berfokus pada spesifikasi fungsional dari perangkat lunak, tester dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program. Pengujian ini dilakukan untuk menguji apakah proses kinerja aplikasi ini sudah sesuai dengan kebutuhan pengguna.
2. Pengujian usabilitas
   Pengujian usabilitas dilakukan dengan cara melakukan survei ke pengguna di sekitar lingkungan Teknik Informatika ITS. Survei dilakukan untuk mengukur tingkat kepuasan aplikasi.

E. Penyusunan laporan tugas akhir
   Pada tahap ini, dilakukan penyusunan laporan yang menjelaskan dasar teori dan metode yang digunakan dalam Tugas Akhir ini serta hasil dari implementasi aplikasi perangkat lunak yang telah dibuat.

1.7 Sistematika Penulisan

Buku Tugas Akhir ini terdiri dari beberapa bab, antara lain sebagai berikut.

BAB I PENDAHULUAN

Bab ini berisi latar belakang masalah, rumusan dan batasan permasalahan, tujuan dan manfaat pembuatan tugas akhir, metodologi yang digunakan, dan sistematika penyusunan tugas akhir.

BAB II TINJAUAN PUSTAKA

Bab ini membahas dasar pembuatan dan beberapa teori penunjang yang berhubungan dengan pokok pembahasan yang mendasari pembuatan tugas akhir ini.

BAB III ANALISIS DAN PERANCANGAN

Bab ini membahas analisis dari sistem yang dibuat meliputi analisis permasalahan, deskripsi umum perangkat lunak, spesifikasi kebutuhan, dan identifikasi pengguna. Kemudian membahas rancangan dari sistem yang dibuat meliputi rancangan skenario kasus penggunaan, arsitektur, data, dan antarmuka.

BAB IV IMPLEMENTASI

Bab ini membahas implementasi dari rancangan sistem yang dilakukan pada tahap perancangan. Penjelasan implementasi meliputi implementasi pembangkitan area aplikasi, dan antarmuka aplikasi.

BAB V PENGUJIAN DAN EVALUASI

Bab ini membahas pengujian dari aplikasi yang dibuat dengan melihat keluaran yang dihasilkan oleh aplikasi dan evaluasi untuk mengetahui kemampuan aplikasi.

BAB VI PENUTUP

Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan serta saran untuk pengembangan aplikasi selanjutnya.

BAB II
TINJAUAN PUSTAKA

Pada bab ini akan dibahas mengenai teori-teori yang menjadi dasar dari pembuatan Tugas Akhir ini. Teori-teori tersebut adalah *Mixed Reality,* Unity3D, Google Cardboard, Deteksi ujung tangan dan Rancang Bangun Perangkat Lunak.

2.1 *Mixed Reality*

Mixed reality (MR) merupakan sub-bagian dari virtual reality. MR adalah teknologi yang menggabungkan dunia nyata dengan dunia virtual di dalam sebuah perangkat [5].

Penggunaan MR telah banyak digunakan, termasuk raksasa perusahaan perangkat lunak Microsoft yang mengeluarkan Microsoft Hololens. Hololens merupakan perangkat pertama yang memungkinkan pengguna untuk berinteraksi dengan dunia hologram tanpa batas [6].

2.2 Unity3D

Unity adalah middleware yang dapat digunakan di Windows, Linux, dan iOS. Aplikasi yang dihasilkan Unity dapat dijalankan di berbagai platform seperti Windows, iOS, Android, WebGL, dan lain sebagainya. Bahasa pemrograman yang dapat digunakan untuk mengembangkan aplikasi di dalam Unity adalah C#, Javascript, dan BOO. Biasanya, Unity3D digunakan untuk membuat game, tapi tidak menutup kemungkinan untuk membuat aplikasi interaktif lain [7].

Unity3D memiliki *scene* yang digunakan sebagai wadah untuk menampung obyek-obyek yang ada dalam aplikasi. Obyek-obyek yang telah terinstansiasi disebut *gameobject*. *Gameobject* tersebut memiliki beberapa *class* bawaan, seperti transform yang memberikan informasi dasar berupa posisi, *localscale*, dan rotasi.

Banyaknya forum diskusi dan referensi lain di internet untuk membantu pengembang aplikasi jika mendapat masalah. *Asset Store*

yang merupakan tempat jual beli asset sehingga pengembang tidak perlu susah untuk mendapatkan modul yang diinginkan.

2.3 Google Cardboard

Google Cardboard merupakan salah satu produk Google yang memungkinkan penggunanya berinteraksi dengan dunia maya. Dengan *software development kit* (SDK) yang disediakan pengembang Google, memungkinkan pengembang aplikasi Unity untuk menciptakan pengalaman interaksi dengan dunia maya [8].

Beberapa fitur yang diberikan adalah pelacakan gerakan kepala pengguna, konfigurasi tampilan *stereo* untuk beberapa model cardboard tertentu, koreksi penyimpangan *gyroscope* yang otomatis, dan lain sebagainya. Penulis merubah sedikit *script* dari Google Cardboard SDK untuk menyesuaikan kecepatan bergerak perangkat android dan kecepatan bergerak *gameobject* cardboard.

2.4 Deteksi ujung tangan

Dalam Tugas Akhir ini, pendeteksian ujung tangan merupakan hal yang utama, karena titik paling ujung tangan diperlukan untuk menggerakkan kontrol *pointer* aplikasi. Metode pengenalan ujung tangan pada tugas akhir ini tidak menggunakan metode yang biasa digunakan. Namun, sekedar cukup untuk mengakomodasi kebutuhan aplikasi.

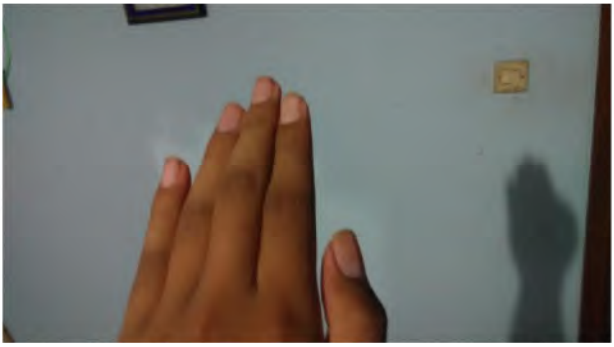
Henry Chang dan Ulises Robles melakukan penelitian mengenai deteksi wajah [10]. Dalam salah satu bab nya, terdapat materi mengenai *Skin Segmentation* (segmentasi tangan).

2.4.1 Konversi RGB ke Grayscale

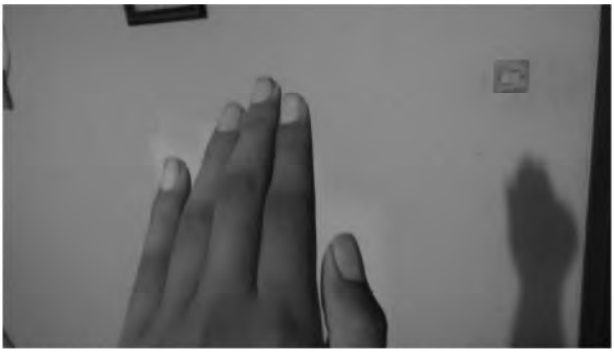
Sebuah gambar yang diambil oleh kamera terdiri atas titik-titik yang memiliki tingkat warna tertentu, titk-titik itu disebut dengan piksel. Setiap piksel memiliki tingkat warna tersendiri tergantung ruang warna yang digunakan. Ruang warna sendiri adalah satuan untuk warna. Terdapat beberapa ruang warna, seperti RGB (*Red, Green, Blue*), CMYK (*Cyan, Magenta, Yellow, Black*),

HSV (*Hue, Saturation, Value*), *Grayscale* (keabuan) dan lain sebagainya.

Gambar yang diambil oleh kamera dalam aplikasi ini menggunakan ruang warna RGB, seperti pada Gambar 2.1. Agar bisa dengan mudah diproses pada tahap selanjutnya, ruang warna RGB harus diubah ke ruang warna *grayscale*, seperti terlihat pada Gambar 2.2. Untuk mengubahnya, cukup merata-rata kan setiap nilai yang terdapat pada R, G, dan B.



Gambar 2.1. Gambar kamera dengan ruang warna RGB



Gambar 2.2. Gambar yang telah dikonversi ke ruang warna *Grayscale*

2.4.2 Nilai *Threshold* yang telah ditentukan

Penelitian Chang dan Robles menggunakan adaptive thresholding yang memiliki input gambar dengan ruang warna g*rayscale* dan memiliki luaran berupa gambar dengan dua warna saja, yaitu hitam dan putih. Untuk menentukan warna tersebut merupakan warna hitam atau putih, Chang dan Robles menggunakan *adaptive thresholding*. Dari penelitiannya, *threshold* yang paling optimal memiliki tingkat keabuan dengan nilai 0.4 dari skala 0 sampai dengan 1 [10]. Namun, nilai *grayscale* yang digunakan dalam Tugas Akhir ini berskala 0 sampai 255. Dengan demikian, nilai 0.4 dalam skala 0 sampai dengan 1 dapat dikonversi menjadi nilai 102 dalam skala 0 sampai dengan 255. Gambar 2.3. menunjukkan hasil *thresholding* yang dinyatakan dalam dua jenis warna saja, yaitu hitam dan putih. Warna hitam (bernilai 0) menunjukkan bahwa piksel tersebut memiliki tingkat keabuan di atas 102, sedangkan warna putih (bernilai 1) merupakan piksel yang memiliki tingkat keabuan di bawah 102.

Gambar 2.3. Gambar dengan dua jenis warna (hitam dan putih)

2.4.3 *Two-Pass (Connected-Component Labelling)*

Gambar 2.3. menunjukkan bahwa terdapat lima bagian berwarna putih. Dengan demikian, sulit bagi program untuk menentukan diantara kelima bagian tersebut untuk dijadikan acuan

dalam menggerakkan pointer. *Two-pass Connected Component Labelling* digunakan untuk memberi label antara piksel-piksel yang berhubungan. Disebut *two-pass* karena memiliki dua tahap dalam menjalankannya.

Tahap pertama digunakan untuk memberikan label sementara setiap piksel serta mencatat label tetangga *4-connectivity*-nya. *4-connectivity* merupakan tetangga-tetangga piksel yang berada di *A(x-1, y-1), B(x, y-1), C(x+1, y-1), dan D(x-1, y)*. Penentuan label sementara diambil dari iterasi setiap piksel yang berwarna putih (bernilai 1) sesuai dengan aturan berikut:

1. Cari label tetangga *4-connectivity*
2. Jika tidak memiliki tetangga bernilai sama, maka buat label baru.
   a. Tambahkan label baru ke daftar label
3. Jika memiliki tetangga bernilai sama, cari label tetangga paling kecil
   a. Tambahkan relasi antara label-label yang bertetangga

Sedangkan pada tahap kedua, digunakan untuk menetapkan kembali label piksel. Daftar label yang dibuat dalam tahap pertama digunakan dalam tahap kedua untuk menggabungkan label yang bertetangga. Label baru yang digunakan merupakan label yang bernilai paling kecil dari daftar label yang bertetangga. Hasil dari masing-masing daerah dan label dapat dilihat pada Gambar 2.4.

Gambar 2.4. Label untuk setiap bagian

(Halaman ini sengaja dikosongkan)

BAB III
ANALISIS DAN PERANCANGAN

Pada bab ini akan dibahas analisis dan perancangan yang akan digunakan untuk menyelesaikan Tugas Akhir ini. Dalam Tugas Akhir ini akan dibuat sebuah aplikasi yang menerapkan teknologi MR dengan menggunakan perangkat Android dan Google Cardboard.

3.1 Analisis Sistem

Cara penataan ruangan dengan cara yang masih konvensional, yaitu dengan melepas dan memasang kembali mebel membutuhkan tenaga yang besar dan waktu yang lama. Penerapan teknologi informasi melalui komputer dan aplikasi pemodelan 3D kurang memberikan kesan nyata dalam penataan ruangan.

Kemajuan teknologi informasi memunculkan inovasi-inovasi baru dalam bidang interaksi manusia dan komputer. Kemajuan teknologi tersebut semakin memanjakan manusia dalam interaksinya dengan komputer, termasuk benda virtual yang ada di dalam komputer. Penggunaan *virtual reality* (VR) *tools* seperti Google dengan Cardboardnya, Facebook dengan Oculusnya, serta Microsoft dengan Hololensnya, membuat interaksi antara manusia dengan benda *vitual* menjadi lebih terasa.

Selain VR, ada pula *augmented reality* (AR) yang memproyeksikan benda *virtual* seperti ada pada dunia nyata. Benda *virtual* yang dihasilkan oleh AR dapat dilihat dengan perangkat tertentu yang memiliki daya penglihatan yang terbatas. Jarak antara mata dengan perangkat ponsel pintar dapat mempengaruhi kemampuan melihat benda *virtual*. Sama seperti lubang pintu, semakin jauh jarak antara perangkat ponsel pintar dengan mata, semakin kecil pula bagian dari benda virtual yang dapat dilihat.

Penataan ruangan yang mengimplementasikan teknologi VR dengan lingkungan asli merupakan inti dari pengerjaan tugas akhir ini. Interaksi manusia dengan benda virtual menggunakan kontrol tangan menambah kesan *immersive* serta dapat mengurangi

penggunaan tenaga pengguna dibandingkan dengan melepas lalu memasang furnitur.

Pembangunan aplikasi ini menggunakan Unity3D yang biasa digunakan untuk membuat aplikasi aplikasi tiga dimensi dan aplikasi VR. Bahasa pemrograman C# merupakan salah satu bahasa pemrograman yang dapat digunakan dalam pembangunan aplikasi menggunakan Unity3D. Penggunaan Cardboard SDK juga dibutuhkan agar dapat dijalankan dalam perangkat Android dan Google Cardboard. Sementara itu, gambar kamera yang ditangkap akan dilakukan segmentasi dan dicari titik tertinggi sebagai kontrol pointer aplikasi.

3.2 Perancangan Aplikasi

3.2.1 Deskripsi Umum Perangkat Lunak

Aplikasi yang dibangun merupakan aplikasi yang menerapkan teknologi MR. Aplikasi ini diimplementasikan ke dalam perangkat Android. Pengguna dapat berinteraksi dengan aplikasi menggunakan Google Cardboard dengan kontrol tangan. Pengguna dapat memilih mebel yang diinginkan lalu menambahkannya untuk dilihat. Pengguna dapat berputar ke kanan, kiri, atas, atau bawah untuk melihat mebel yang sudah ditambahkan. Jika ada mebel yang tidak sesuai, pengguna dapat menghapusnya. Kontrol aplikasi menggunakan pointer yang mendeteksi tangan dengan titik piksel tertinggi.

Menu awal hanya memiliki satu pilihan, yaitu menambahkan obyek mebel/*furniture*. Setelah memilih untuk menambahkan obyek, maka akan muncul daftar obyek yang sudah terdaftar. Dengan memilih salah satu *thumbnail* obyek, maka aplikasi akan menampilkan detail mebel. Pengguna dapat menambahkan obyek mebel dengan mengarahkan pointer kontrol ke icon "+".

Jika obyek mebel yang tidak sesuai dengan keinginan, maka pengguna dapat menghapus dengan cara mengarahkan pointer kontrol ke obyek yang diinginkan dan menunggu hingga menu *pop*

up muncul, lalu arahkan kontrol pointer ke ikon "x" untuk menghapus obyek mebel.

Dengan menggunakan aplikasi ini, akan menambah kesan *immersive* kepada pengguna mengenai *virtual reality* dan *augmented reality*. Selain itu, penggunaan tenaga untuk menata ruang juga dapat berkurang.

3.2.2 Spesifikasi Kebutuhan Fungsional

Berdasarkan deskripsi umum sistem, kebutuhan fungsional aplikasi ini adalah dapat melihat obyek *virtual* pada lingkungan nyata dapat berinteraksi dengan obyek *virtual* tersebut.

3.2.3 Spesifikasi Kebutuhan Non-Fungsional

Terdapat beberapa kebutuhan non-fungsional yang apabila dipenuhi, dapat meningkatkan kualitas dari aplikasi ini. Berikut daftar kebutuhan non-fungsional:

1. **Realtime**

   Aplikasi ini menyajikan tampilan realtime yang diambil melalui kamera belakang perangkat Android. Perangkat Android akan mengambil gambar yang nyata lalu diproses dan ditampilkan di layar perangkat tersebut. Gambar yang diambil tidak memiliki resolusi yang besar karena akan memengaruhi FPS (*framerate per second*).

2. **Model obyek virtual tiga dimensi**

   Model obyek virtual yang digunakan diadaptasi dari obyek-obyek nyata yang sering terdapat di rumah. Untuk itu, kesan *immersive* semakin bertambah karena menggunakan obyek yang asli.

3.2.4 Karakteristik Pengguna

Berdasarkan deskripsi umum di atas, maka dapat diketahui bahwa pengguna yang akan menggunakan aplikasi ini ada dua jenis,

yaitu orang awam dan desainer ruangan. Karakteristik pengguna tercantum dalam Tabel 3.1.

Tabel 3.1 Karakteristik Pengguna

Nama Aktor	Tugas	Hak Akses Aplikasi	Kemampuan yang harus dimiliki
Pengguna	Pihak yang menggunakan	Menggunakan aplikasi	Tidak ada

3.3 Perancangan Sistem

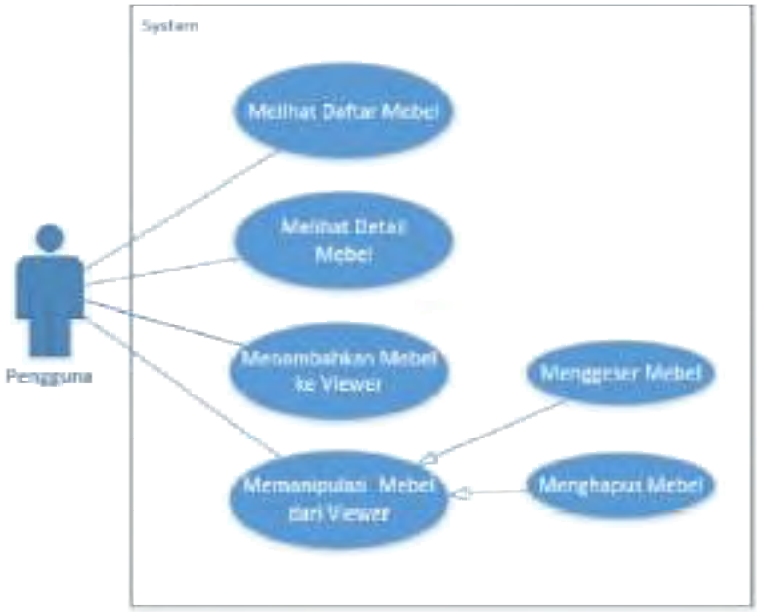
Tahap perancangan dalam subbab ini dibagi menjadi beberapa bagian yaitu perancangan diagram kasus penggunaan, perancangan skenario kasus penggunaan, perancangan antarmuka, perancangan kontrol aplikasi, dan perancangan diagram alur aplikasi.

3.3.1 Perancangan Diagram Kasus Penggunaan

Dalam aplikasi Tugas Akhir ini, terdapat dua kasus penggunaan, antara lain memilih karakter dan bermain. Pengguna atau entitas luar dari sistem adalah pemain.

3.3.2 Perancangan Skenario Kasus Penggunaan

Diagram kasus penggunaan yang terdapat di dalam sistem ditunjukkan pada Gambar 3.1. Diagram kasus penggunaan tersebut memiliki empat kebutuhan fungsional utama, yaitu melihat daftar mebel, melihat detail mebel, menambahkan mebel ke *viewer*, dan manipulasi mebel dari *viewer*.



Gambar 3.1 Diagram kasus aplikasi

Penjelasan dari masing-masing kasus penggunaan dicantumkan pada Tabel 3.2. Tabel tersebut berisi penjelasan skenario yang akan dilakukan ketika pengujian.

Tabel 3.2 Tabel scenario kasus penggunaan

No	Kode Kasus Penggunaan	Nama Kasus Penggunaan	Keterangan
1	UC-001	Melihat Daftar Mebel	Pengguna dapat melihat daftar mebel yang tersedia dalam aplikasi
2	UC-002	Melihat Detail Mebel	Pengguna dapat meliha detail mebel yang terpilih

3	UC-003	Menambahkan Mebel ke Viewer	Pengguna dapat menambahkan mebel yang diinginkan untuk dilihat di Viewer
4	UC-004	Memanipulasi Mebel dari Viewer	Pengguna dapat memanipulasi (menghapus atau mengubah posisi) mebel yang terpilih dari Viewer

3.3.2.1 Kasus Penggunaan Aplikasi

Penjelasan kasus penggunaan aplikasi untuk skenario UC-001 yakni Memilih karakter ditunjukkan pada Tabel 3.3.

Tabel 3.3 Skenario Kasus Penggunaan Melihat Daftar Mebel

Nama Kasus Penggunaan	Melihat Daftar Mebel
Kode	UC-001
Deskripsi	Kasus penggunaan dimana pengguna melihat daftar mebel yang tersedia di dalam aplikasi
Aktor	Orang Awam
Kondisi Awal	-
Alur Normal	1. Pengguna memilih tombol "Menu" ("|"), 2. Sistem menampilkan menu utama, 3. Pengguna memilih tombol "Add" ("+"), 4. Sistem menampilkan daftar mebel

Selanjutnya, penjelasan kasus penggunaan aplikasi untuk skenario UC-002 yakni melihat detail mebel ditunjukkan pada Tabel 3.4.

Tabel 3.4 Skenario Kasus Penggunaan Melihat Detail Mebel

Nama Kasus Penggunaan	Melihat Detail Mebel
Kode	UC-002
Deskripsi	Pengguna dapat melihat informasi yang lebih detail dari mebel yang diinginkan
Aktor	Orang Awam

Kondisi Awal	Pengguna sudah melakukan kasus penggunaan UC-001
Alur Normal	1. Pengguna memilih salah satu ikon mebel yang diinginkan 2. Sistem menampilkan menampilkan preview mebel beserta informasi detail mebel tersebut

Sedangkan untuk UC-003, yaitu Menambahkan Mebel ke Viewer dapat dilihat pada Tabel 3.5.

Tabel 3.5 Skenario Kasus Penggunaan Menambahkan Mebel ke Viewer

Nama Kasus Penggunaan	Menambahkan Mebel ke Viewer
Kode	UC-003
Deskripsi	Kasus penggunaan yang memungkinan pengguna untuk menambahkan mebel yang diinginkan ke dalam Viewer.
Aktor	Orang Awam
Kondisi Awal	Pengguna sudah melakukan kasus penggunaan UC-002
Alur Normal	1. Pengguna memilih salah satu mebel yang diinginkan 2. Sistem menampilkan detail mebel 3. Pengguna memilih tombol "Add" ("+"), 4. Sistem menambahkan mebel yang diinginkan ke Viewer

Skenario kasus penggunaan yang terakhir (UC-004), yaitu Memanipulasi Mebel dari Viewer dapat dilihat pada Tabel 3.6.

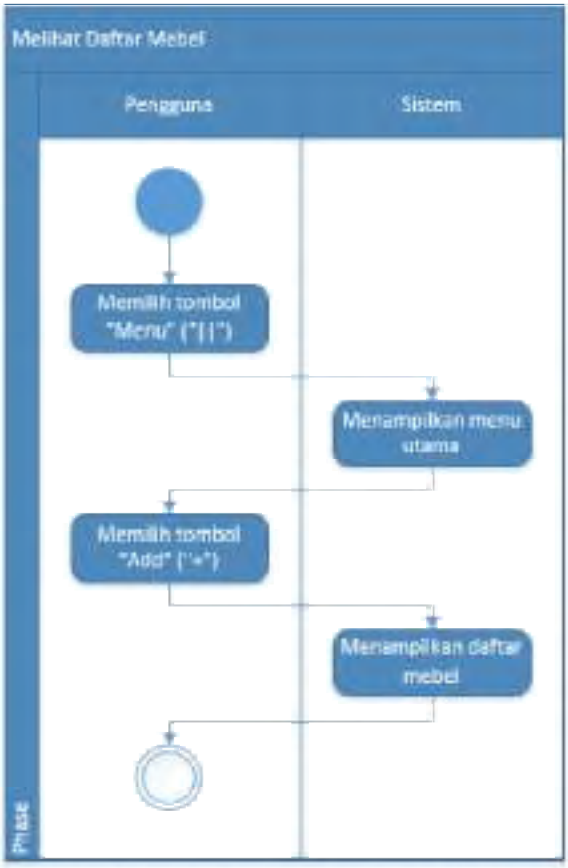
Tabel 3.6 Skenario Kasus Penggunaan Memanipulasi Mebel dari Viewer

Nama Kasus Penggunaan	Memanipulasi Mebel dari Viewer
Kode	UC-004

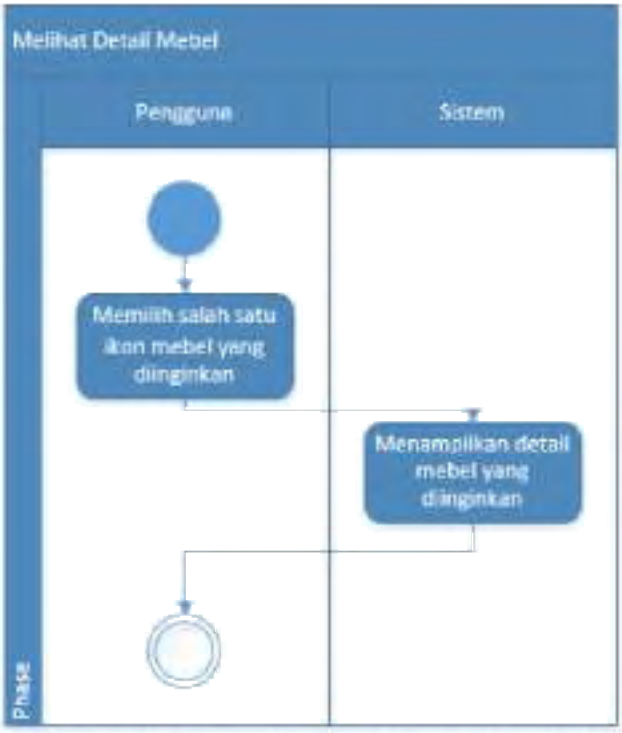
Deskripsi	Pengguna dapat memanipulasi (mengubah posisi dan menghapus) mebel yang diinginkan
Aktor	Orang Awam
Kondisi Awal	Mebel yang ingin dimanipulasi terdapat di dalam Viewer
Alur Normal	1. Pengguna memilih mebel yang ingin dihapus 2. Sistem menampilkan menu pop-up dan menggerakkan mebel sesuai dengan posisi pointer 3. Pengguna memilih tombol ConfirmMove 4. Sistem merubah posisi mebel
Alur Alternatif	3.1.1. Pengguna memilih tombol CancelMove 3.1.2. Sistem mengembalikan posisi mebel ke posisi awal 3.2.1. Pengguna memilih tombol DeleteObject 3.2.2. Sistem menghapus mebel yang terpilih

3.3.2.2 Diagram Aktivitas

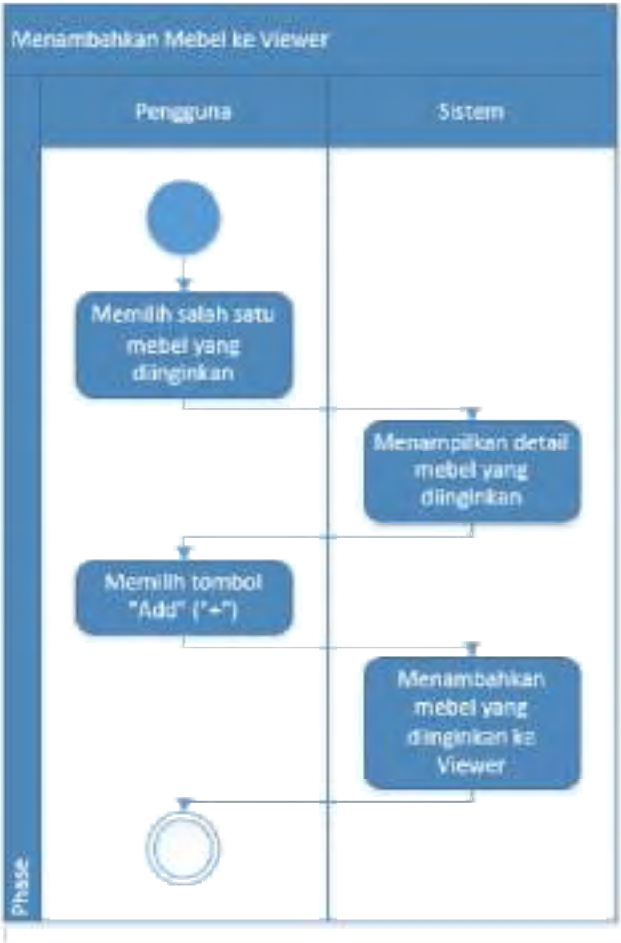
Diagram aktivitas menampilkan langkah-langkah normal yang harus dilakukan pemain untuk menjalankan studi kasus aplikasi dimulai dari awal aplikasi hingga kondisi akhir. Gambar-gambar di bawah ini merupakan gambar-gambar diagram aktivitas masing-masing kasus penggunaan. Untuk kasus penggunaan UC-001, yaitu melihat daftar mebel dapat dilihat pada Gambar 3.2. Gambar 3.3. merupakan gambar diagram aktivitas untuk kasus penggunaan UC-002, melihat detail mebel. Sedangkan untuk kasus penggunaan UC-003 dapat dilihat pada Gambar 3.4. Kasus penggunaan UC-003 sendiri dilakukan untuk menambah mebel yang diinginkan pengguna ke dalam *Viewer*. Sedangkan untuk kasus penggunaan UC-004 dapat dilihat pada Gambar 3.5. Kasus penggunaan UC-004 sendiri merupakan kasus penggunaan yang merupakan generalisasi dari menggeser mebel dan menghapus mebel. Untuk itu, kasus penggunaan ini merupakan kasus penggunaan dengan alur yang paling panjang dan paling kompleks dalam aplikasi penataan ruangan ini.



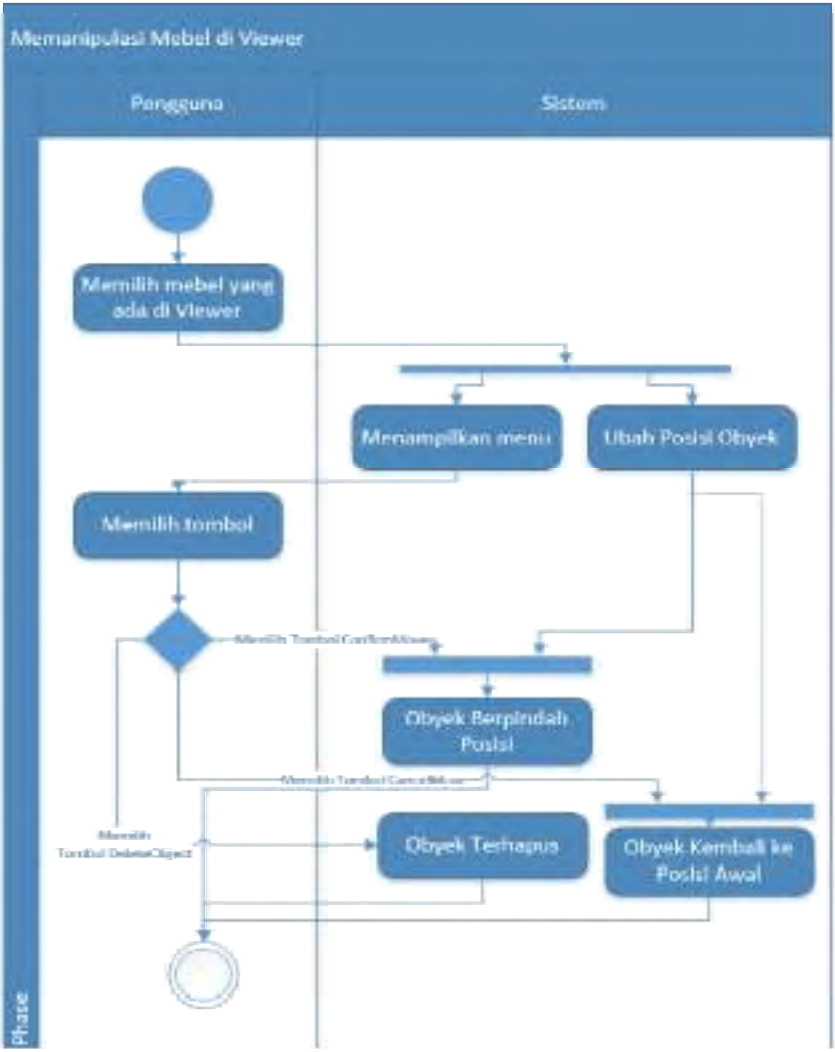
Gambar 3.2 Diagram aktivitas Memilih Daftar Mebel



Gambar 3.3 Diagram aktivitas Melihat Detail Mebel



Gambar 3.4 Diagram aktivitas Menambahkan Mebel ke Viewer



Gambar 3.5 Diagram aktivitas Memanipulasi Mebel dari Viewer

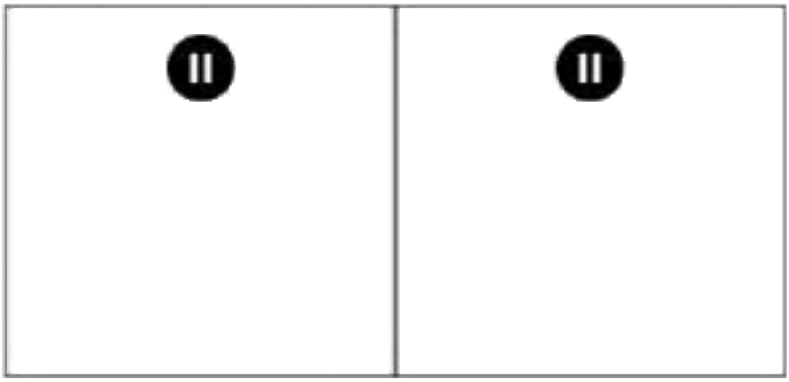
3.3.3 Perancangan Antarmuka Pengguna

Subbab ini membahas rancangan antarmuka pengguna yang akan digunakan pada Tugas Akhir ini. Rancangan antarmuka yang dibahas meliputi ketentuan masukan dan rancangan jendela antarmuka.

Pada awal masuk aplikasi, pengguna hanya akan melihat Viewer yang menampilkan gambar lingkungan nyata yang diambil oleh kamera perangkat Android. Namun, di dalam tombol *Pause* ("||") terdapat tombol-tombol yang lainnya, seperti Menu utama, daftar mebel, detail mebel, dan menu *pop up* yang akan muncul ketika ada mebel yang terpilih.

3.3.3.1 Antarmuka *Viewer*

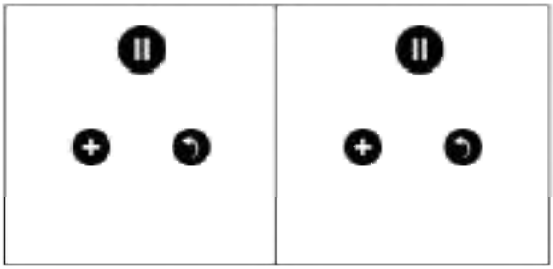
Antarmuka *Viewer* merupakan antarmuka pertama yang muncul saat membuka aplikasi setelah *splashscreen*. *Splashscreen* sendiri merupakan tampilan awal dimana pengembang "memamerkan" nama produk, tim pembuat, *engine* yang digunakan, dan lain sebagainya. Pada antarmuka ini, hanya terdapat satu tombol, yaitu *Pause* ("||") yang akan menampilkan tombol lain. Rancangan antarmuka *viewer* dapat dilihat pada gambar 3.6.



Gambar 3.6 Antarmuka *Viewer*

3.3.3.2 Antarmuka Menu Utama

Antarmuka ini tidak jauh berbeda dari antarmuka *Viewer*. Antarmuka ini merupakan menu *pop up* yang muncul jika pengguna memilih tombol *Pause*. Pada antarmuka ini, terdapat dua pilihan tombol lain, yaitu tombol *add* ("+") dan tombol *back*. Tombol *add* digunakan untuk menampilkan daftar mebel yang ada pada aplikasi, sedangkan tombol *back* digunakan untuk kembali ke viewer.

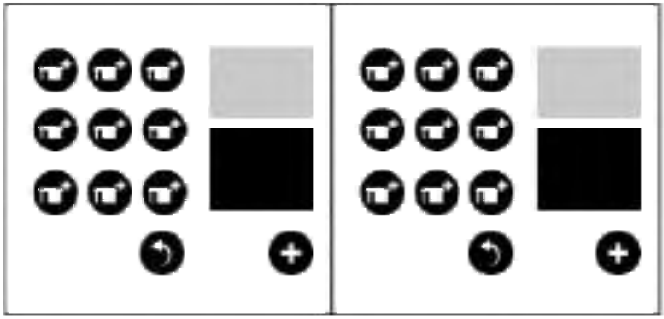


Gambar 3.7 Antarmuka Menu Utama

3.3.3.3 Antarmuka Daftar Mebel

Antarmuka ini juga masih di dalam *Viewer*. Layar seperti terbagi menjadi dua, kiri dan kanan. Bagian kiri terdiri dari daftar mebel yang tersedia, sedangkan bagian kanan berisi tiga komponen utama. Komponen pertama yaitu *preview* mebel yang tergambar seperti kotak paling atas, komponen kedua yang berupa kotam hitam dengan tulisan putih yang merupakan deskripsi mebel. Serta komponen ketiga yang merupakan tombol untuk menambahkan mebel tersebut di *Viewer*. Perancangan antarmuka daftar mebel dapat dilihat pada Gambar 3.8.

Untuk melihat deskripsi mebel, cukup pilih salah satu mebel yang diinginkan. Deskripsi dan detail yang ditampilkan berupa nama, merek, harga, ukuran, dan warna.

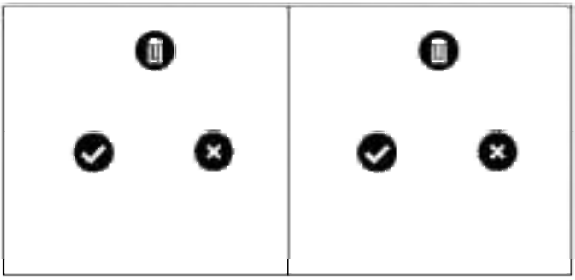


Gambar 3.8 Antarmuka Daftar Meubel

3.3.3.4 Antarmuka Mebel Terpilih

Mebel-mebel yang telah tertambahkan di *Viewer* dapat dimanipulasi oleh pengguna. Manipulasi mebel yang dimaksud adalah memindah dan dihapus. Dengan terpilihnya sebuah mebel, maka dalam kondisi ini mebel dapat dipindah. Antarmuka yang ditampilkan adalah menu pop up yang dapat dilihat pada Gambar 3.9. Terdapat tiga tombol, yaitu *Remove*, *Confirm*, dan *Cancel*.

Apabila pengguna memilih tombol *Remove*, maka mebel yang terpilih akan dihapus dari *Viewer*. Jika pengguna memilih tombol *Confirm*, maka mebel akan ditempatkan di posisi tersebut. Sedangkan jika pengguna memilih tombol *Cancel*, maka mebel kembali diletakkan di posisi awal dan menu *pop up* akan hilang.



Gambar 3.9 Antarmuka Mebel Terpilih

(Halaman ini sengaja dikosongkan)

BAB IV
IMPLEMENTASI

Pada bab ini akan dibahas mengenai implementasi dari perancangan perangkat lunak. Di dalamnya mencakup proses penerapan dan pengimplementasian algoritma, dan antar muka yang mengacu pada rancangan yang telah dibahas sebelumnya.

4.1 Lingkungan Implementasi

Lingkungan implementasi perangkat lunak memiliki spesifikasi minimal seperti yang ditunjukkan tabel 4.1.

Tabel 4.1 Lingkungan Implementasi Perangkat Lunak

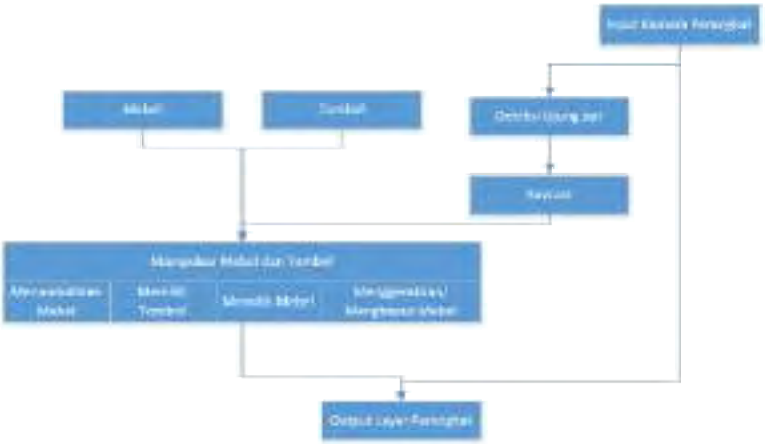
Perangkat Keras	Prosesor: Qualcomm MSM8228 Snapdragon Quad Core 1.6 GHz Cortex-A7 Memori: 1024MB RAM Resolusi Kamera : 8 Megapixel Sensor : *GyroSensor*
Perangkat Lunak	Sistem Operasi: Android Kitkat

4.2 Implementasi Alur Proses Aplikasi

Pada subbab ini, akan dibahas implementasi alur proses aplikasi yang telah dibangun pada bab sebelumnya. Gambaran umum bagaimana aplikasi bekerja dapat dilihat dalam Gambar 4.1.

4.2.1 Implementasi Mengambil Kamera dari Perangkat

Kamera perangkat digunakan untuk mengambil gambar lingkungan sekitar dan tangan pengguna. Gambar tersebut akan ditampilkan ke layar sebagai background atau lingkungan aplikasi. Sedangkan gambar tangan akan diproses lebih lanjut agar pengguna dapat menggerakkan pointer aplikasi.



Gambar 4.1 Cara Kerja Aplikasi

Prosesor dan memori yang dimiliki perangkat mempengaruhi resolusi gambar yang diambil oleh kamera. Resolusi kamera tersebut memengaruhi kecepatan proses pendeteksian ujung jari. Dengan demikian, resolusi gambar yang diambil memiliki kualitas yang tidak bagus namun cukup mengakomodasi kebutuhan, yaitu berukuran 320 piksel x 240 piksel. Implementasi untuk menangkap gambar dari kamera perangkat dapat dilihat pada Kode Sumber 4.1. Sedangkan implementasi untuk menampilkan gambar ke layar perangkat dalat dilihat pada Kode Sumber 4.2.

```
1.  void Update () {
2.      WebCamTexture webCamTexture = new WebCamTexture (3
    20, 240);
3.      HandTracking ht = GetComponent<HandTracking> ();
4.      ht.UpdateFrame (webCamTexture.GetPixels32 ());
5.      webCamTexture.Play ();
6.  }
```

Kode Sumber 4.1. Mengambil gambar dari kamera perangkat

```
1. void Update () {
2.     Texture2D tex = new Texture2D(webCamTexture.width, w
   ebCamTexture.height);
3.     tex.SetPixels32(ht.GetFrame("main"));
4.     tex.Apply();
5. }
```

Kode Sumber 4.2. Menampilkan gambar ke layar perangkat

4.2.2 Implementasi Deteksi Ujung Tangan

Cara untuk mengoperasikan aplikasi adalah menggunakan *pointer*. *Pointer* yang dimaksud adalah penunjuk yang digunakan untuk mengarahkan keiinginan pengguna ke tombol atau mebel yang diinginkan.

Posisi *pointer* didapatkan dari pendeteksian ujung tangan. Deteksi ujung tangan sendiri telah dijelaskan pada Bab II, Tinjauan Pustaka. Implementasi dilakukan dengan mengadaptasi *Two-Pass Filter* sesuai kebutuhan aplikasi.

Proses yang pertama kali adalah menentukan wilayah gambar yang hanya memiliki dua jenis warna saja. Penetuan warna menggunakan nilai yang tetap, yaitu 102 seperti yang ada pada Bab II. Untuk piksel yang memiliki nilai di bawah 102 mendapatkan label 0. Itu menandakan bahwa piksel tersebut tidak memiliki label. Sedangkan piksel dengan nilai di atas 102 akan diberikan nilai -1 sebagai tanda bahwa piksel tersebut merupakan salah satu piksel dari bagian tertentu. Impelementasi dapat dilihat pada Kode Sumber 4.3.

```
1.  public void FirstStep()
2.      {
3.          initialLabel = new int[frameLength];
4.          for (int a = 0; a < frameLength; a++)
5.          {
6.              if (WhichIsWhite(a))
7.              {
```

```
                initialLabel[a] = -1;
            }
            else
            {
                initialLabel[a] = 0;
            }
        }
    }
```

Kode Sumber 4.3. Implementasi pemberian label awal

Dalam kode sumber di atas, tejadi pemanggilan fungsi *WhichIsWhite(parameter)*. Fungsi tersebut merupakan fungsi yang menentukan apakah piksel tersebut bernilai di atas 102 atau tidak. Fungsi *WhichIsWhite(parameter)* akan mengembalikan nilai *True* jika piksel tersebut bernilai kurang dari 102, dan mengembalikan nilai *False* jika piksel tersebut bernilai lebih dari 102. Implementasi fungsi *WhichIsWhite (parameter)* dapat dilihat pada Kode Sumber 4.4.

```
public bool WhichIsWhite(int index){
    int gray = GrayColor(index);
    if (gray < 100)
        return true;
    else
        return false;
}
```

Kode Sumber 4.4. Implementasi fungsi WhichIsWhite

Setelah itu, masuk ke tahap pertama yaitu pemberian label awal dengan kode sumber yang dapat dilihat di Kode Sumber 4.5. Tahap ini merupakan adaptasi dari *Two-Pass Connected-Component Labelling* tahap pertama. Setiap piksel akan diiterasi dan jika ditemukan piksel berlabel bukan 0, tetangga piksel tersebut akan dilakukan pengecekan seperti yang dijelaskan pada Bab II. Jika ternyata piksel tersebut tidak memiliki tetangga berlabel bukan 0,

maka pada piksel tersebut akan dibuatkan label baru. Sedangkan jika terdapat tetangga yang berlabel bukan 0, maka label untuk piksel tersebut bernilai tetangga yang berlabel paling kecil. Selain itu, juga dilakukan pencatatan relasi antar label.

```
1.  public void SecondStep()
2.  {
3.
4.      for (int a = 0; a < frameHeight; a++)
5.      {
6.          for (int b = 0; b < frameWidth; b++)
7.          {
8.              if (initialLabel[GetIndexPoint(b, a)] == 0
    )
9.                  initialLabel[GetIndexPoint(b, a)] = 0;

10.             else
11.             {
12.                 int[] neighborsIndex = new int[4];
13.                 for (int x = 0; x < neighborsIndex.Len
    gth; x++) {
14.                     neighborsIndex[x] = -1;
15.                 }
16.                 // TopLeft
17.                 if (b-1 > 0 && a-1 > 0)
18.                     neighborsIndex[0] = GetIndexPoint(
    b - 1, a - 1);
19.                 // Top
20.                 if (a-1 > 0)
21.                     neighborsIndex[1] = GetIndexPoint(
    b, a - 1);
22.                 // TopRight
23.                 if (b+1 < frameWidth && a-1 > 0)
24.                     neighborsIndex[2] = GetIndexPoint(
    b + 1, a - 1);
25.                 // Left
26.                 if (b-1 > 0)
27.                     neighborsIndex[3] = GetIndexPoint(
    b - 1, a);
28.
29.                 int neighborsWhiteCount = 0;
```

```
30.                 for (int i = 0; i < neighborsIndex.Len
    gth; i++)
31.                 {
32.                     if (neighborsIndex[i] > 0 && neigh
    borsIndex[i] < frameLength)
33.                     {
34.                         if (initialLabel[neighborsInde
    x[i]] != 0)
35.                             neighborsWhiteCount++;
36.                     }
37.                 }
38.
39.                 if (neighborsWhiteCount == 0)
40.                 {
41.                     // Add New Label
42.                     List<int> newLabel = new List<int>
    ();
43.                     int newLabelValue = labelAndRelati
    on.Count + 1;
44.                     newLabel.Add(newLabelValue);
45.                     labelAndRelation.Add(newLabel);
46.                     initialLabel[GetIndexPoint(b, a)]
    = newLabelValue;
47.                 }
48.                 else
49.                 {
50.                     // Get Lowest Label
51.                     ConnectLabel(neighborsIndex);
52.                     initialLabel[GetIndexPoint(b, a)]
    = LowestNeighbor(neighborsIndex);
53.                 }
54.             }
55.         }
56.     }
57. }
```

Kode Sumber 4.5. Langkah pertama *Two-Pass Connected-Component Labelling*

Terjadi pemanggilan fungsi *ConnectLabel (neighbors)* pada kode sumber 4.5. Fungsi tersebut berguna untuk membuat relasi

antar label yang bertetangga. Implementasi fungsi tersebut dapat dilihat pada Kode Sumber 4.6. Dalam fungsi ini, dilakukan pengecekan untuk setiap label tetangga. Jika hubungan antara label tetangga belum terhubung, maka fungsi tersebut akan menambahkan tetangga label *x* ke dalam list *y*, begitu pula sebaliknya.

```
1.  public void ConnectLabel(int[] neighborsIndex)
2.  {
3.      for (int a = 0; a < neighborsIndex.Length; a++)
4.      {
5.          for (int b = 0; b < neighborsIndex.Length; b++
    )
6.          {
7.              int sameNumberCounter = 0;
8.              if (neighborsIndex[a] > 0 && initialLabel[
    neighborsIndex[a]] > 0 && neighborsIndex[b] > 0)
9.              {
10.                 for (int c = 0; c < labelAndRelation[i
    nitialLabel[neighborsIndex[a]] - 1].Count; c++)
11.                 {
12.                     int initialLabelValue = initialLab
    el[neighborsIndex[b]];
13.                     int labelAndRelationValue = labelA
    ndRelation[initialLabel[neighborsIndex[a]] - 1][c];
14.                     if (labelAndRelationValue == initi
    alLabelValue)
15.                         sameNumberCounter++;
16.                 }
17.                 if (sameNumberCounter == 0 && initialL
    abel[neighborsIndex[b]] > 0)
18.                     labelAndRelation[initialLabel[neig
    hborsIndex[a]] -
     1].Add(initialLabel[neighborsIndex[b]]);
19.             }
20.         }
21.     }
22. }
```

Kode Sumber 4.6. Implementasi fungsi ConnectLabel

Selain fungsi *ConnectLabel(neighbors)*, fungsi lain yang dipanggil adalah fungsi *LowestNeighbor(neighbors)*. Fungsi tersebut berguna untuk menentukan tetangga yang memiliki label terkecil. Implementasi fungsi *LowestNeighbor (neighbors)* dapat dilihat pada Kode Sumber 4.7.

```
1.  public int LowestNeighbor(int[] neighborsIndex)
2.  {
3.      int lowestNeighbor = frameLength;
4.      int lowestNeighborIndex = 0;
5.      for (int i = 0; i < neighborsIndex.Length; i++)
6.      {
7.          if (neighborsIndex[i] > 0 && neighborsIndex[i]
    < frameLength)
8.          {
9.              if (lowestNeighbor > neighborsIndex[i] &&
    initialLabel[neighborsIndex[i]] > 0)
10.             {
11.                 lowestNeighbor = initialLabel[neighbor
    sIndex[i]];
12.                 lowestNeighborIndex = neighborsIndex[i
    ];
13.             }
14.         }
15.     }
16.     return lowestNeighbor;
17. }
```

Kode Sumber 4.7. Fungsi LowestNeighbors(neighbors)

Aplikasi ini melewatkan tahap kedua dari algoritma *Two-Pass Connected-Component Labelling* karena dapat mempercepat jalannya aplikasi. Aplikasi membutuhkan bagian dengan jumlah piksel terbanyak, untuk itu pemberian label ulang dapat diganti dengan langsung menentukan bagian dengan piksel terbanyak. Implementasinya dapat dilihat pada Kode Sumber 4.8. Fungsi tersebut berjalan dua kali, yang pertama adalah pembuatan jumlah piksel awal masing-masing label. Masing-masing label memiliki luas awal 0. Sedangkan aliran kedua berguna untuk menghitung

jumlah piksel. Setiap piksel akan diiterasi, jika piksel tersebut memiliki label bukan 0, maka setiap tetangga label tersebut akan dihitung pula.

```
1.  public void ThirdStep()
2.  {
3.      areas = new List<int>();
4.      for (int a = 0; a < labelAndRelation.Count; a++)
5.      {
6.          areas.Add(0);
7.      }
8.
9.      for (int a = 0; a < frameLength; a++)
10.     {
11.         if (initialLabel[a] > 0)
12.         {
13.             for (int b = 0; b < labelAndRelation[initi
    alLabel[a] - 1].Count; b++)
14.             {
15.                 areas[labelAndRelation[initialLabel[a]
     - 1][b] - 1]++;
16.             }
17.         }
18.     }
19. }
```

Kode Sumber 4.8. Pencarian jumlah piksel setiap label

Setelah proses itu dijalankan, fungsi yang selanjutnya dipanggil adalah fungsi *FourthStep*(). Fungsi *FourthStep*() berguna untuk menentukan bagian yang memiliki jumlah piksel paling banyak. *Variable areas* yang menampung jumlah piksel masing-masing label akan diiterasi untuk menentukan bagian yang memiliki jumlah paling banyak. *Variable* yang menampung daftar jumlah piksel perlabel berjenis *List of integer* dikarenakan jumlah label tidak sama untuk setiap *frame* pengambilan gambar oleh kamera Implementasi fungsi *FourthStep*() dapat dilihat pada Kode Sumber 4.9.

```
1.  public void FourthStep()
2.  {
3.      largestAreaRelation = new List<int>();
4.      largestArea = 0;
5.      for (int a = 0; a < areas.Count; a++)
6.      {
7.          if (areas[a] > areas[largestArea])
8.              largestArea = a;
9.      }
10.     largestAreaRelation = labelAndRelation[largestArea
    ];
11. }
```

Kode Sumber 4.9. Pencarian label dengan jumlah piksel terbanyak

Dengan ditemukannya label dengan jumlah piksel terbanyak, dapat ditentukan pula titik ujung tangannya. Titik ujung tangan ditentukan dengan cara iterasi setiap piksel dari paling belakang sampai paling depan. Alasan mengapa iterasi dimulai dari titik paling belakang akan dijelaskan pada paragraf selanjutnya. Proses ini akan mengembalikan sebuah nilai berupa index dari array piksel gambar. Implementasi fungsi dapat dilihat pada Kode Sumber 4.10.

```
1.  public int FifthStep()
2.  {
3.      for (int a = frameHeight-1; a >= 0; a--)
4.      {
5.          for (int b = 0; b < frameWidth; b++)
6.          {
7.              if (initialLabel[GetIndexPoint(b,a)] > 0)

8.              {
9.                  for (int c = 0; c < labelAndRelation[l
    argestArea].Count; c++)
10.                 {
11.                     if (initialLabel[GetIndexPoint(b,a
    )] == labelAndRelation[largestArea][c])
12.                         return GetIndexPoint(b,a);
13.                 }
```

```
14.                 }
15.             }
16.         }
17. }
```

Kode Sumber 4.10. Pencarian titik ujung jari

Iterasi berawal dari piksel dengan deret belakang. Hal itu dilakukan karena perbedaan pengurutan array. *Array* piksel gambar yang diambil oleh kamera merupakan *array* satu dimensi. Sedangkan proses-proses di atas menggunakan *array* dua dimensi. Selain itu, apabila array satu dimensi tersebut langsung dikonversi menjadi *array* dua dimensi, titik (0, 0) berada pada pojok kiri atas. Sedangkan titik (0,0) pada diagram kartesius berada di pojok kiri bawah. Kode Sumber 4.11. dan Kode Sumber 4.12. berfungsi untuk mengonversi *index* dari *array* satu dimensi yang diambil kamera ke titik *x* dan *y*. Sedangkan Kode Sumber 4.13. berguna untuk mengonversi titik *x* dan *y* kembali ke angka *index*.

```
1.  int GetXPoint(int a){
2.      return a%frameWidth;
3.  }
```

Kode Sumber 4.11. Fungsi untuk mendapatkan titik *x* dari *index* array

```
1.  int GetYPoint(int a){
2.      return frameHeight - (a/frameWidth) - 1;
3.  }
```

Kode Sumber 4.12. Fungsi untuk mendapatkan titik *y* dari *index* array

```
1.  public int GetIndexPoint(int x, int y){
2.      return y * frameWidth + x;
3.  }
```

Kode Sumber 4.13. Fungsi untuk mendapatkan index dari titik *x* dan *y*

4.2.3 *RayCasting*

Dalam Unity3D, tersedia fitur *raycast*. *Raycast* adalah sinar *virtual* yang dipancarkan dari sebuah koordinat tiga dimensi yang memiliki panjang tertentu. Seperti yang diketahui, titik hasil proses deteksi ujung tangan berupa titik dua dimensi, namun aplikasi membutuhkan koordinat tiga dimensi agar obyek dapat terseleksi. Untuk itu, *raycast* mengonversi titik dua dimensi menjadi titik tiga dimensi dengan memancarkan sinar sejauh *z*. Benda-benda yang dilewati raycast dapat memanggil fungsi *Action*(). Fungsi *Action*() berbeda-beda setiap obyeknya dan dijelaskan pada sub-bab selanjutnya. Sedangkan implementasi raycast dapat dilihat pada Kode Sumber 4.14.

```
1.  void RayCasting(Vector3 position){
2.      Ray rayOrigin = imp.rayOrigin;
3.      RaycastHit[] rayInfo = Physics.RaycastAll(rayOrigi
    n);
4.      gm.RayCasting (rayInfo, position);
5.  }
```

Kode Sumber 4.14. Implementasi *raycast*

4.2.4 Implementasi *Viewer*

Viewer merupakan tempat dimana setiap *GameObject* yang ada di aplikasi ini berada. Pada antarmuka ini, hanya terdapat satu tombol, yaitu *Pause* ("||") yang akan menampilkan tombol lain. Implementasi antarmuka viewer dapat dilihat pada gambar 4.2.

Gambar 4.2 Implementasi *Viewer*

Tombol Pause digunakan untuk memanggil menu *pop up* lain, yaitu Menu Utama Implementasi tombol *Pause* dapat dilihat pada Gambar 4.3. Sedangkan implementasi kode sumber untuk tombol Pause ditunjukkan pada Kode Sumber 4.15. Sedangkan kode sumber tombol *Add* dapat Dilihat pada Kode Sumber 4.16.

Gambar 4.3 Tombol *Pause*

```
1.public class ButtonPause : MonoBehaviour, IButton{
2.     public GameObject mainMenu;
```

```
3.
4.     void Start () {
5.         if (mainMenu == null)
6.             mainMenu = GameObject.Find ("Main");
7.     }
8.
9.     public void Action(){
10.      mainMenu.SetActive (mainMenu.activeSelf ? false :
  true);
11.      }
12. }
```

Kode Sumber 4.15 Tombol *Pause*

4.2.5 Implementasi Menu Utama

Menu utama berisi dua buah tombol utama, yaitu tombol *Add* dan tombol *Back*. Kedua tombol tersebut muncul seperti *pop up*. Implementasi antarmuka menu utama dapat dilihat pada Gambar 4.4.

Gambar 4.4 Implementasi Antarmuka Menu Utama

Di dalam aplikasi ini, terdapat dua buah jenis tombol *Add*, yaitu tombol *Add* untuk melihat daftar mebel dan tombol *Add* untuk

menambahkan mebel ke dalam *Viewer*. Tombol *Add* berguna untuk melihat daftar Ikon untuk tombol *Add* dapat dilihat pada Gambar 4.5.

Gambar 4.5 Ikon Tombol *Add*

```
1.  void Start () {
2.      if (insertMenu == null)
3.          insertMenu = GameObject.Find ("Insert");
4.  }
5.  public void Action(){
6.      insertMenu.SetActive (true);
7.      this.transform.parent.transform.gameObject.SetActi
    ve (false);
8.  }
```

Kode Sumber 4.16 Implementasi Tombol *Add*

Selain tombol *Add*, terdapat tombol *Back* yang berfungsi untuk kembali ke status selanjutnya. Ikon tombol *Back* dapat dilihat pada Gambar 4.6. Sedangkan kode sumber untuk tombol *Back* dapat dilihat pada Kode Sumber 4.17.

Gambar 4.6 Ikon Tombol *Back*

```
1.  public void Action(){
```

```
2.      this.transform.parent.gameObject.SetActive (false)
    ;
3.  }
```

Kode Sumber 4.17 Implementasi Tombol Back

4.2.6 Implementasi Daftar Mebel dan Detail Mebel

Pengguna dapat melihat daftar mebel yang tersedia dalam aplikasi. Terdapat 12 mebel dengan 3 jenis berbeda, yaitu kursi, meja, dan perlengkapan lainnnya. Pengguna dapat melihat daftar mebel yang ada seperti yang ditunjukkan pada Gambar 4.7. Dalam gambar tersebut, ada beberapa komponen utama, seperti ikon mebel, *Preview Section*, *Information Section*, tombol *Add*, dan tombol *Back*.

Gambar 4.7 Implementasi Antarmuka Daftar Mebel

Ikon mebel berfungsi untuk mengetahui mebel mana yang akan dimasukkan ke dalam Viewer. Daftar mebel beserta ikonnya dapat dilihat pada Tabel 4.2. Model-model dalam tabel tersebut diambil dari penyedia model online gratis yang memiliki alamat http://3dsky.org/.

Tabel 4.2 Tabel Daftar Mebel dan Detailnya

Gambar	Jenis	Nama Benda	Nama Obyek
	Kursi	Kursi 1	Meubel0
	Kursi	Kursi 2	Meubel1
	Kursi	Kursi 3	Meubel2
	Kursi	Kursi 4	Meubel3
	Lemari	Lemari 1	Meubel4
	Lemari	Lemari 2	Meubel5
	Lemari	Lemari 3	Meubel6

	Lemari	Lemari 4	Meubel7
	Meja	Meja 1	Meubel8
	Meja	Meja 2	Meubel9
	Meja	Meja 3	Meubel10
	Meja	Meja 4	Meubel11

Untuk masing-masing ikon mebel, memiliki fungsi yang sama untuk mengubah *Preview Section* dan *Information Section* dalam script *Show Detail*. Kode sumber dapat dilihat pada Kode Sumber 4.18.

```
1.  public DetailModel detailModel;
2.  public void Action(){
3.      detailModel.ShowDetail (this.name);
4.  }
```

Kode Sumber 4.18 Implementasi *Show Detail*

Masing-masing ikon memiliki akhiran nama (*postfix*) berupa id atau index mebel. Id tersebut digunakan untuk menentukan mebel mana yang akan ditampilkan pada *Preview Section* dan *Information Section*.

Preview Section merupakan daerah dimana *preview* mebel ditampilkan. *Preview Section* akan berubah dari Gambar ke Gambar

ketika salah satu ikon mebel terpilih. Selain itu, *Information* Section juga akan menampilkan informasi mebel jika salah satu mebel terpilih, seperti yang terlihat pada Gambar 4.8.

Gambar 4.8 *Preview* dan *Information Section* sebelum dipilih

Untuk mengganti *Preview Section* dan *Information Section* dengan mebel yang dipilih, *ShowDetail(parameter)* memanggil script yang ada pada Kode Sumber 4.19. Dari *objectToAdd* yang tersedia, dipilih obyek yang memiliki index sama dengan *postfix parameter* yang dikirim.

```
1.  public GameObject preview;
2.  public GameObject infoText;
3.  public GameObject buttonAdd;
4.  public Texture[] thumbs = new Texture[12];
5.  public Texture defaultThumbs;
6.  public GameObject[] objectToAdd = new GameObject[12];
7.
8.  TextMesh tmInfoText;
9.  ButtonAdd ba;
10.
11. void Start () {
12.     tmInfoText = infoText.GetComponent<TextMesh> ();
```

```
13.     ba = buttonAdd.GetComponent<ButtonAdd> ();
14.     preview.GetComponent<Renderer> ().material.mainTex
    ture = defaultThumbs;
15. }
16. public void ShowDetail(string name){
17.     int currentModel = int.Parse(name.Substring (5));

18.     preview.GetComponent<Renderer> ().material.mainTex
    ture = thumbs [currentModel];
19.     tmInfoText.text = PlayerPrefs.GetString ("Info-
    " + name);
20.     ba.SetTargetModel (objectToAdd[currentModel]);
21. }
```

Kode Sumber 4.19 Kode Sumber *Preview* dan *Information Section*

Dengan adanya tombol *Add*, maka mebel yang terpilih akan ditampilkan ke dalam *Viewer*. Ikon untuk tombol *Add* sendiri dapat dilihat pada Gambar 4.9. Sedangkan kode sumber untuk tombol tersebut dapat dilihat pada Kode Sumber 4.20.

Gambar 4.9 Ikon *Add*

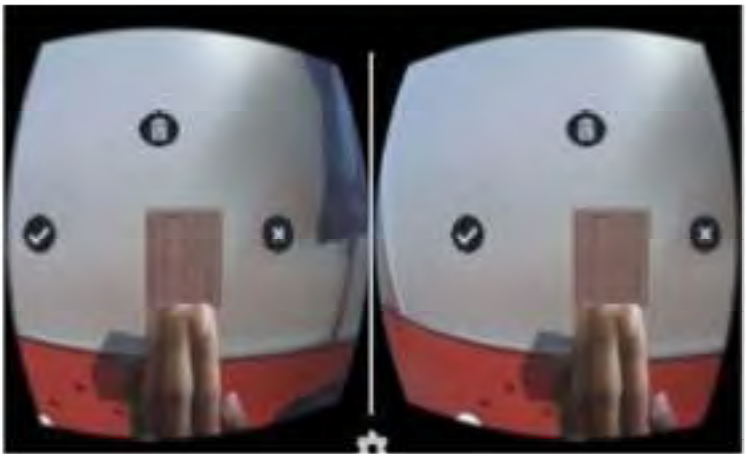
```
1.  public GameObject objectToAdd;
2.  public Transform spawnTransform;
3.
4.  public void SetTargetModel(GameObject theObject){
5.      objectToAdd = theObject;
6.  }
7.  public void Action(){
```

```
8.      GameObject go = Instantiate (objectToAdd, spawnTra
    nsform.position, objectToAdd.transform.rotation) as Ga
    meObject;
9.      go.name = go.name.Substring(0, go.name.Length -
     7);
10.     this.transform.parent.transform.parent.gameObject.
    SetActive (false);
11. }
```

Kode Sumber 4.20 Kode Sumber *Add*

4.2.7 Implementasi Memilih Obyek

Memilih obyek memiliki kondisi jika ada mebel dalam *Viewer*. Pemilihan obyek bertujuan untuk memindahkan obyek atau menghapus obyek. Pilihan menu akan muncul seperti Gambar 4.10. jika salah satu obyek terpilih.



Gambar 4.10 Antarmuka Implementasi Memilih Obyek

Terdapat tiga buah tombol *pop up* yang muncul ketika salah satu obyek terpilih, yaitu *ConfirmMove*, *CancelMove*, dan *DeleteObyek*. Ikon Tombol *ConfirmMove* dapat dilihat pada Gambar 4.11.

Gambar 4.11 Ikon *ConfirmMove*

Kegunaan tombol *ConfirmMove* adalah untuk mengonfirmasi bahwa posisi obyek tersebut diubah ke posisi yang diinginkan pengguna dengan menggunakan Kode Sumber 4.21. Tombol yang selanjutnya adalah *CancelMove*, ikon tombol *CancelMove* dapat dilihat pada Gambar 4.12.

```
1.  public GameManager gm;
2.  public void Action(){
3.      gm.onMovingObject = null;
4.      this.transform.parent.gameObject.SetActive (false)
    ;
5.      GameObject.Find ("Menus").GetComponent<Menus> ().p
    auseButton.SetActive (true);
6.  }
```

Kode Sumber 4.21 Implementasi Tombol *ConfirmMove*

Gambar 4.12 *CancelMove*

Tombol *CancelMove* bisa dipilih pengguna apabila pengguna ingin untuk mengembalikan mebel ke tempat semula sebelum

dipindah. Tombol *CancelMove* akan menjalankan fungsi yang dapat dilihat pada Kode Sumber 4.22.

```
1.  ModelProperty model;
2.  public GameManager gm;
3.
4.  void Start () {
5.      model = gm.GetComponent<GameManager>().selectedMod
    el.GetComponent<ModelProperty>();
6.  }
7.
8.  public void Action(){
9.      Vector3 startPosition = model.startPosition;
10.     model.transform.position = startPosition;
11.     gm.onMovingObject = null;
12.     this.transform.parent.gameObject.SetActive (false)
    ;
13.     GameObject.Find ("Menus").GetComponent<Menus> ().p
    auseButton.SetActive (true);
14. }
```

Kode Sumber 4.22 *CancelMove*

Tombol yang terakhir ada pada antarmuka ini adalah *DeleteObject*. Ikon untuk tombol *DeleteObject* dapat dilihat pada Gambar 4.13. Sedangkan kode sumber untuk tombol *DeleteObject* dapat dilihat pada Kode Sumber 4.23.

Gambar 4.13 Ikon *DeleteObject*

```
1.  public void Action(){
2.          Destroy(GameObject.Find ("GameManager").GetCom
    ponent<GameManager>().selectedModel.gameObject);
3.          this.transform.parent.gameObject.SetActive (fa
    lse);
4.          GameObject.Find ("Menus").GetComponent<Menus>
    ().pauseButton.SetActive (true);
5.      }
```

Kode Sumber 4.23 Implementasi *DeleteObject*

BAB V
PENGUJIAN DAN EVALUASI

Pada bab ini akan dijelaskan mengenai rangkaian uji coba dan evaluasi yang dilakukan. Proses pengujian dilakukan menggunakan metode *black-box* berdasarkan skenario yang telah ditentukan dan pengujian dilakukan dengan survei langsung kepada pengguna.

5.1 Lingkungan Uji Coba

Lingkungan pelaksanaan uji coba meliputi perangkat keras dan perangkat lunak yang akan digunakan pada sistem ini. Spesifikasi perangkat keras dan perangkat lunak yang digunakan dalam rangka uji coba perangkat lunak ini ditunjukkan pada Tabel 5.1.

Tabel 5.1 Lingkungan Uji Coba Perangkat Lunak

Nama Perangkat	Xiaomi Redmi 2 Prime
Perangkat Keras	Prosesor: Quad-core 1.2 GHz Cortex-A53 Memori: 2GB RAM Kamera : 8 MP Layar : 4.7 Inchi
Perangkat Lunak	Sistem Operasi: Android Kitkat

5.2 Skenario Pengujian Fungsionalitas

Pengujian ini dilakukan untuk menguji apakah fungsionalitas yang diidentifikasi pada tahap kebutuhan benar-benar diimplementasikan dan bekerja semestinya. Selain itu, langkah ini ditujukan untuk mengetahui kesesuaian keluaran dari setiap tahapan atau langkah penggunaan fitur terhadap skenario yang dipersiapkan. Pengujian dilakukan dengan menggunakan metode *black-box*.

5.2.1 Skenario Pengujian Melihat Daftar Mebel

Skenario pengujian untuk melihat daftar mebel digunakan untuk mengetahui apakah fungsionalitas melihat daftar mebel sudah berjalan sesuai dengan apa yang diharapkan. Scenario ini ditunjukkan pada Tabel 5.2.

Tabel 5.2 Pengujian Melihat Daftar Mebel

	UC-001
Kondisi Awal	-
Prosedur Pengujian	Pada tampilan awal viewer, pengguna memilih tombol *Pause* "‖" lalu menekan tombol *Add* "+".
Hasil yang diharapkan	Pengguna dapat melihat daftar mebel yang tersedia dalam aplikasi
Hasil yang diperoleh	Pengguna dapat melihat daftar mebel yang tersedia dalam aplikasi
Kesimpulan.	Pengujian berhasil

5.2.1.1 PF-01: Pengujian Melihat Daftar Mebel

Pengujian dimulai saat aplikasi pertama kali dibuka. Setelah itu, pengguna harus memilih tombol *Pause*, lalu memilih tombol *Add* untuk melihat daftar mebel yang tersedia dalam aplikasi. Antarmuka yang ditampilkan dapat dilihat pada Gambar 5.1.

Gambar 5.1 Hasil Pengujian Melihat Daftar Mebel

5.2.2 Skenario Pengujian Melihat Detail Mebel

Skenario pengujian untuk melihat detail mebel dilakukan untuk mengetahui apakah fungsionalitas dalam melihat detail mebl dapat berjalan sesuai keiinginan. Skenario ini ditunjukkan pada Tabel 5.3.

Tabel 5.3 Pengujian Melihat Detail Mebel

	UC-002
Kondisi Awal	Terdapat daftar mebel yang tersedia dalam aplikasi
Prosedur Pengujian	Pengguna memilih salah satu ikon mebel yang tersedia.
Hasil yang diharapkan	Pengguna dapat melihat detail mebel pada *Preview Section* dan *Information Section*.
Hasil yang diperoleh	Pengguna dapat melihat detail mebel pada *Preview Section* dan *Information Section*.
Kesimpulan.	Pengujian berhasil

5.2.2.1 PF-02: Pengujian Melihat Detail Mebel

Pengujian dimulai setelah pengguna menyelesaikan scenario UC-001. Setelah daftar mebel tampil, pengguna dapat memilih salah satu mebel untuk dilihat detailnya. Antarmuka yang ditampilkan dapat dilihat pada Gambar 5.2.

Gambar 5.2 Hasil Pengujian Melihat Detail Mebel

5.2.3 Skenario Pengujian Menambahkan Mebel ke *Viewer*

Skenario pengujian untuk menambahkan mebel ke *viewer* bertujuan untuk mengetahui apakah fungsionalitas menambah mebel ke *viewer* berhasil. Skenario ini ditunjukkan pada Tabel 5.4.

Tabel 5.4 Pengujian Menambah Mebel ke Viewer

	UC-003
Kondisi Awal	Telah terpilih salah satu mebel yang diinginkan
Prosedur Pengujian	Pengguna memilih tombol *Add* pada bagian kanan
Hasil yang diharapkan	Pengguna dapat melihat mebel yang diinginkan muncul di *Viewer*

Hasil yang diperoleh	Pengguna dapat melihat mebel yang diinginkan muncul di *Viewer*
Kesimpulan.	Pengujian berhasil

5.2.3.1 PF-03: Pengujian Menambahkan Mebel ke Viewer

Setelah salah satu mebel terpilih, maka akan muncul detail mebel seperti yang ada pada UC-002. Dengan demikian, pengguna dapat menambahkan mebel ke *Viewer* melalui tombol *Add* yang tersedia di bagian kanan menu. Antarmuka yang ditampilkan dapat dilihat pada Gambar 5.3.

Gambar 5.3 Hasil Pengujian Menambahkan Mebel ke *Viewer*

5.2.4 Skenario Pengujian Memanipulasi Mebel dari *Viewer*

Skenario pengujian untuk memanipulasi mebel dari *Viewer* bertujuan untuk melihat apakah fungsionalitas untuk manipulasi mebel dari *Viewer* berhasil . Skenario ini ditunjukkan pada Tabel 5.5.

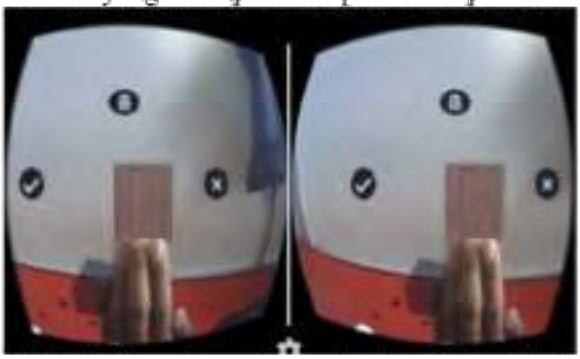
Tabel 5.5 Pengujian Memanipulasi Mebel dari Viewer

	UC-004
Kondisi Awal	Telah terpilih salah satu mebel yang

	diinginkan di dalam Viewer
Prosedur Pengujian	Pengguna memilih tombol *ConfirmMove* untuk meletakkan mebel ke posisi yang diinginkan. Sedangkan pengguna dapat memilih tombol *DeleteObject* untuk menghapus mebel yang diinginkan
Hasil yang diharapkan	Mebel yang diinginkan termanipulasi oleh pengguna
Hasil yang diperoleh	Mebel yang diinginkan termanipulasi oleh pengguna
Kesimpulan.	Pengujian berhasil

5.2.4.1 PF-04: Pengujian Memanipulasi Mebel dari Viewer

Pengguna memilih mebel yang diingkan lalu muncul menu *pop up* yang berisi tombol *ConfirmMove* untuk mengubah posisi ke tempat yang diinginkan, lalu tombol *DeleteObject* untuk menghapus mebel, dan *CancelMove* untuk mengembalikan mebel ke tempat semula. Antarmuka yang ditampilkan dapat dilihat pada Gambar 5.4.



Gambar 5.4 Hasil Pengujian Memanipulasi Mebel dari Viewer

5.2.5 Skenario Pengujian dengan *Background* yang Kompleks

Batasan masalah aplikasi ini adalah latar belakang aplikasi yang berwarna putih atau memiliki tingkat keabuan diatas 50%.

Sehingga, pengujian dengan latar belakang yang kompleks dan tidak sesuai batasan dapat mengacaukan pendeteksian ujung jari, sehingga pointer yang ditampilkan tidak sesuai keinginan pengguna. Gambar 5.5 menunjukkan pointer yang salah dalam pendeteksian ujung jari.

Gambar 5.5. Kesalahan Pendeteksian Pointer

5.2.5.1 Hasil Pengujian Fungsional

Subbab ini berisi tentang hasil pengujian fungsionalitas yang sudah dilakukan berdasarkan pada PF01, PF02, PF03 dan PF04. Empat pengujian yang telah dilakukan menunjukkan bahwa semua fungsionalitas aplikasi berjalan dengan baik dan sesuai dengan apa yang diharapkan serta sesuai dengan skenario dan alur yang telah dibuat pada perancangan. Rekapitulasi hasil pengujian fungsionalitas ditunjukkan pada Tabel 5.6.

Tabel 5.6 Rekapitulasi Pengujian Fungsional

No	Kode Pengujian	Hasil Pengujian
1.	PF01	Berhasil
2.	PF02	Berhasil
3.	PF03	Berhasil
4.	PF04	Berhasil

5.3 Pengujian Pengguna

Pengujian pada perangkat lunak yang dibangun tidak hanya dilakukan pada fungsionalitas yang dimiliki, tetapi juga pada pengguna untuk percobaan secara langsung. Pengujian ini berfungsi sebagai pengujian subjektif yang bertujuan untuk mengetahui tingkat keberhasilan aplikasi yang dibangun dari sisi pengguna. Hal ini dapat dicapai dengan meminta penilaian dan tanggapan dari pengguna terhadap sejumlah aspek perangkat lunak yang ada.

5.3.1 Skenario Uji Coba Pengguna

Dalam melakukan pengujian perangkat lunak, penguji diminta untuk mencoba menggunakan perangkat lunak yang bersangkutan untuk mencoba semua fungsionalitas dan fitur yang tersedia. Sebelum menguji coba aplikasi yang dibuat dalam tugas akhir ini, penguji terlebih dahulu mencoba merancang ruangan menggunakan aplikasi desktop yang tersedia.

Pengujian aplikasi oleh pengguna dilakukan dengan sebelumnya memberikan informasi seputar aplikasi, kegunaan, dan fitur-fitur yang dimiliki. Setelah informasi tersampaikan, pengguna kemudian diarahkan untuk langsung mencoba aplikasi dengan spesifikasi lingkungan yang sama dengan yang telah diuraikan pada uji coba fungsionalitas.

Jumlah pengguna yang terlibat dalam pengujian perangkat lunak sebanyak tiga orang. Dalam melakukan pengujian, pengguna melakukan percobaan lebih dari satu kali penggunaan untuk masing-masing pengguna.

Dalam memberikan penilaian dan tanggapan, penguji diberikan formulir pengujian perangkat lunak. Formulir pengujian perangkat lunak ini memiliki beberapa aspek penilaian dan pada bagian akhir terdapat saran untuk perbaikan fitur.

5.3.2 Daftar Penguji Perangkat Lunak

Pada subbab ini ditunjukkan daftar pengguna yang bertindak sebagai penguji coba aplikasi yang dibangun. Daftar nama penguji

aplikasi beserta perangkat yang diguakan ditunjukkan pada Tabel 5.7.

Tabel 5.7 Daftar Nama dan Jenis Perangkat Penguji

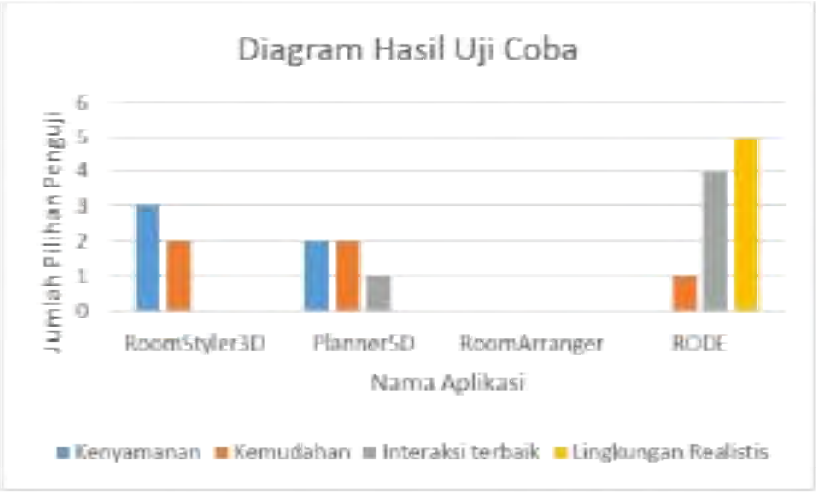
No	Nama	Pekerjaan	Jenis Perangkat
1	Ig. Abraham S.	Mahasiswa Teknik Informatika ITS	Xiaomi Redmi 2 Prime
2	Wahyu W.	Mahasiswa Teknik Informatika ITS	Xiaomi Redmi 2 Prime
3	Yunan Helmi M.	Mahasiswa Teknik Informatika ITS	Xiaomi Redmi 2 Prime
4	Rizkifika A.I	Mahasiswa Teknik Informatika ITS	Xiaomi Redmi 2 Prime
5	Ardha P.S.	Mahasiswa Teknik Informatika ITS	Xiaomi Redmi 2 Prime

5.3.3 Hasil Uji Coba Pengguna

Uji coba yang dilakukan terhadap beberapa pengguna memiliki beberapa aspek yang dipisahkan berdasarkan pengalaman menggunakan aplikasi permodelan 3D, kenyamanan, dan waktu pencapaian sistem.

5.3.3.1 Hasil pembandingan aplikasi desktop dengan aplikasi yang dibuat dalam tugas akhir ini.

Penilaian ini bertujuan untuk membandingkan aplikasi yang pernah digunakan penguji yang diberi nama RODE (*Room Design*). Selain itu, pengalaman penguji juga dapat digunakan untuk menunjukkan bahwa RODE memiliki keunggulan dan kelemahan dalam beberapa hal. Hasil yang didapatkan ditunjukkan pada grafik yang ditunjukkan pada Gambar 5.6. yang menunjukkan perbandingan preferensi penguji dalam beberapa aspek aplikasi.



Gambar 5.6 Diagram Batang Hasil Uji Coba Perbandingan Aplikasi

5.3.3.2 Hasil Penilaian Kenyamanan

Penilaian kenyamanan difokusikan pada penilaian penguji terhadap kemudahan dan kenyamanan dalam memakai perangkat Google Cardboard. Tabel 5.8. menunjukkan nilai masing-masing poin.

Tabel 5.8 Nilai Pengujian

Skala	Keterangan
1	Sangat Tidak Setuju
2	Tidak Setuju
3	Ragu-Ragu
4	Setuju
5	Sangat Setuju

Sedangkan hasil penilaian uji kenyamanan ditunjukkan pada Tabel 5.9.

Tabel 5.9 Hasil Pengujian Kenyamanan

No	Pertanyaan	Jumlah pilihan peserta				
		1	2	3	4	5
1	Saya merasa nyaman dalam menggunakan aplikasi ini	0	1	2	2	0
2	Saya merasa mudah dalam menjalankan task/aktivitas yang diberikan	0	1	2	2	0
3	Aplikasi ini membantu menata ruangan sesuai kondisi asli ruangan	0	1	0	4	0
4	Penggunaan aplikasi ini mengurangi penggunaan energi saya daripada melepas-pasang mebel di ruangan saya	0	0	3	2	0
5	Penggunaan Google VR lebih interaktif dibandingkan aplikasi desktop	0	0	1	4	0
Total		0	3	8	14	0

5.3.3.3 Hasil Penilaian Waktu Pencapaian Sistem

Penilaian ini bertujuan untuk mengukur tingkat pencapaian pengguna. Tabel 5.10. menunjukkan kolom beserta tugas yang harus dilakukan penguji. Peneliti memberikan patokan waktu pencapaian

rata-rata setelah menguji coba sebanyak lima kali. Hasil dari uji coba peneliti dapat dilihat pada Tabel 5.11. Sedangkan pada Tabel 5.12. menunjukkan hasil pengujian untuk masing-masing penguji.

Tabel 5.10 Tugas yang Harus Dijalankan

Nomor	Tugas
1	Melihat daftar mebel
2	Melihat informasi detail mebel
3	Menambahkan mebel ke Viewer
4	Memindahkan mebel
5	Menghapus mebel

Tabel 5.11 Waktu Pencapaian Peneliti

No	Uji Coba	Waktu Pencapaian (menit:detik) per tugas				
		1	2	3	4	5
1	Uji Coba ke-1	18.6	11.5	2:7	7.6	8:0
2	Uji Coba ke-2	17:2	4:5	3:2	23:9	12:1
3	Uji Coba ke-3	19.4	7:9	4:1	18:3	9:2
4	Uji Coba ke-4	20.3	13:5	3:5	14:2	13:0
5	Uji Coba ke-5	17:9	19:0	4:3	10:1	9:2
Rata-Rata		18:12	10:48	3:12	14:24	10:12

Tabel 5.12 Waktu Pencapaian per Tugas

No	Penguji	Waktu Pencapaian (menit:detik) per tugas				
		1	2	3	4	5
1	Ig. Abraham	4:05	36:8	7:4	36:9	35:3
2	Wahyu W.	17:2	7:1	8:5	19:8	20:9

3	Yunan Helmi M.	37:5	7:0	7:1	24:9	26:5
4	Rizkifika A.I	31:6	17:4	9:7	31:5	39:4
6	Ardha P.S.	43:9	20:2	10:9	35:2	50:3

5.3.3.4 Hasil Kolom Saran Penguji

Dalam pengujian tugas akhir ini, terdapat kolom saran yang berisi saran-saran dari penguji mengenai aplikasi tugas akhir ini. Daftar saran masing-masing penguji dapat dilihat pada tabel 5.13. Saran penguji tersebut dapat dijadikan acuan untuk mengembangkan aplikasi hingga pada akhirnya layak untuk di sebarkan ke masyarakat luas.

Tabel 5.13 Tabel Saran Penguji

No	Penguji	Saran
1	Ig. Abraham S.	FPSnya ditingkatan
2	Wahyu W.	Kontrol jangan hanya menggunakan jari karena menjadi capek
3	Yunan Helmi M.	Kurang *realtime*
4	Rizkifika A.I	Lebih baik di-deploy ke *device* yang lebih bagus
5	Ardha P. S.	Kontrol menggunakan jari bisa lebih dijadikan lebih nyaman, karena sebentar saja sangat merasa lelah dan juga perangkat dibuat leibh nyaman untuk pengguna. Interaksi dengan bundaran putih (*pointer*) rasanya kurang cocok karena kurang jelas maksud dan tujuannya.

5.4 Evaluasi Hasil Uji Coba

Dari uji coba fungsional, aplikasi tidak dapat mendeteksi ujung jari dengan latar belakang yang tidak putih atau memiliki tingkat keabuan di atas 50%. Sehingga, *pointer* kontrol aplikasi memiliki koordinat yang salah dan tidak sesuai dengan keinginan penguna.

Penggunaan RODE masih dirasa kurang nyaman oleh penguji. Saran-saran mengenai FPS (*Frame Per Second*) yang kecil, kurang *realitme*, dan *deploy* ke perangkat yang lebih bagus merupakan isu utama dalam pembuatan aplikasi ini. Walaupun perangkat yang digunakan saat pengujian dapat dijalankan, isu FPS yang kecil merupakan isu yang merupakan isu utama dalam RODE. Perancangan dan implementasi aplikasi yang hanya menggunakan bantuan *middleware* Unity3D merupakan alasan yang paling cocok. Dalam pengembangang selanjutnya, penggunaan *framework* atau *library* dapat sangat membantu pengembang untuk mengatasi isu tersebut.

Di sisi lain, ide utama aplikasi sangat baik karena merupakan cara baru dalam interaksi manusia dengan komputer. Dari hasil uji coba yang dilakukan, aplikasi ini unggul dalam hal interaksi terbaik dan lingkungan yang realistis.

Waktu pencapaian penguji dalam menguji coba aplikasi memiliki waktu pencapaian yang berbeda-beda. Waktu tempuh setiap tugas yang dilakukan masing-masing penguji memiliki catatan waktu pencapaian yang lebih lama daripada catatan waktu pencapaian peneliti. Hal itu disebabkan karena penguji masih mencoba aplikasi RODE untuk yang pertama kalinya. Sedangkan penulis telah berkali-kali mencoba aplikasi RODE, sehingga waktu pencapaian lebih singkat karena telah terbiasa menggunakan aplikasi.

BAB VI
KESIMPULAN DAN SARAN

Bab ini membahas mengenai kesimpulan yang dapat diambil dari tujuan pembuatan perangkat lunak dan hasil uji coba yang telah dilakukan sebagai jawaban dari rumusan masalah yang dikemukakan.

6.1. Kesimpulan

Dalam proses pengerjaan Tugas Akhir ini, mulai dari tahap analisis, desain, implementasi, hingga pengujian didapatkan kesimpulan sebagai berikut:

1. Dari pengujian fungsional, dapat disimpulkan bahwa Unity dapat mengambil gambar dari perangkat lalu menampilkannya ke layar. Dengan demikan, semua skenario pengujian dapat dilakukan dengan baik.
2. Dari hasil uji coba aplikasi dapat disimpulkan bahwa aplikasi dapat melakukan pendeteksian ujung jari untuk menggerakkan pointer dengan latar belakang yang putih atau memiliki tingkat keabuan diatas 50%. Dengan demikian, pengguna dapat memanipulasi obyek tiga dimensi sesuai skenario yang ditentukan. Latar belakang yang memiliki tingkat keabuan diatas 50% membuat aplikasi sulit mengenali ujung jari sehingga memberikan koordinat yang tidak tepat pada *pointer*.
3. Penggunaan RODE dirasa kurang nyaman karena faktor FPS yang kecil dan kurang *realtime*.
4. RODE dapat dijalankan dalam perangkat yang berspesifikasi minimum, namun lebih baik dijalankan di perangkat yang lebih bagus.
5. Waktu pencapaian penguji dalam menjalankan tugas lebih lambat daripada waktu pencapaian penulis.

Selain kesimpulan, juga terdapat saran yang ditujukan untuk pengembangan perangkat lunak lebih lanjut.

6.2. Saran

Berikut adalah beberapa saran untuk pengembangan sistem di masa yang akan datang berdasarkan pada hasil perancangan, implementasi dan uji coba yang telah dilakukan.

1. Penggunaan *library* yang sudah ada seperti OpenCV for Unity dapat membantu pengembang dalam mengembangkan aplikasi. Selain itu, penggunaan OpenCV for Unity juga dapat mengurangi beberapa batasan sistem dalam aplikasi ini. Selain itu, penggunaan *library* dapat mempercepat FPS aplikasi
2. Penggunaan jaringan internet untuk mengunduh mebel terbaru dapat memperkaya jumlah mebel.
3. Penggunaan metode pengenalan tangan dapat mempermudah pengguna dalam menggerakkan pointer dan menyeleksi obyek pada *Viewer*

DAFTAR PUSTAKA

[1] A. R. K. C. Wasim Ahmed Khan, "*Virtual Manufacturing*", "Springer Series in Advanced Manufacturing", 2011.

[2] T. H. Taehee Lee, "*Handy AR : Markerless Inspection of Augmented Reality Objects Using Fingertip Tracking*", 2009.

[3] M. Kölsch, "*Vision Based Hand Gesture Interfaces for Wearable Computing and Virtual Environments*", 2004.

[4] F. S. D. V. K. H. Gerd Bruder, "*Immersive Virtual Studio for Architectural Exploration*", 2009.

[5] Milgram Paul, Kishino Fumio, *"A Taxonomy of Mixed Reality Visual Displays"*, "IEICE Transactions on Information Systems", Vol E77-D, No.12 December 1994.

[6] Microsoft, "Hololens", 2016 [Online]

[7] Unity Technologies, "Unity", 2015 [Online]

[8] Google Inc, "Google Developers", 2015 [Online]

[9] Elgammal Ahmed, Muang Crystal, H Dunxu, *"Skin Detection - a Short Tutorial"*, 2009

[10] Chang H, Robles U, "Face Detection", "Skin Segmentation", 200 [Online]

(Halaman ini sengaja dikosongkan)

BIODATA PENULIS

M. Ghozie Manggala Lahir di Ponorogo, 22 April 1994. Penulis menghabiskan masamasa sekolahnya (SD, SMP, SMA) di Ponorogo. Saat ini melanjutkan berkuliah di Institut Teknologi Sepuluh Nopember Surabaya. Tepatnya di S1 Teknik Informatika Jurusan Teknik Informatika Institut Teknologi Sepuluh Nopember.